1997

Factoring integers defined by second and third order recurrence relations

Kirk Ocke

Rochester Institute of Technology
Computer Science Department

Factoring Integers Defined by Second and Third Order Recurrence Relations.

by
Kirk Ocke

A thesis, submitted to
The Faculty of the Computer Science Department
in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science

Approved by:

Dr. Stanisław Radziszowski
Rochester Institute of Technology

Dr. Peter Anderson
Rochester Institute of Technology

Dr. Pasquale Arpaia
Saint John Fisher College

December 8, 1997

Factoring Integers Defined by Second and Third Order Recurrence Relations.

I, Kirk J. Ocke, hereby **grant permission** to the Wallace Library of the Rochester Institute of Technology to reproduce my thesis in whole or in part. Any reproduction will not be for commercial use or profit.

Kirk J. Ocke

December 10, 1997

Abstract

Factoring the first two-hundred and fifty Fibonacci numbers using just trial division would take an unreasonable amount of time. Instead the problem must be attacked using modern factorization algorithms. We look not only at the Fibonacci numbers, but also at factoring integers defined by other second and third order recurrence relations. Specifically we include the Fibonacci, Tribonacci and Lucas numbers. We have verified the known factorizations of first 382 Fibonacci numbers and the first 185 Lucas numbers, we also completely factored the first 311 Tribonacci numbers.

Contents

1 A Brief Thank you **3**

2 Introduction **4**
2.1 Definitions, Notation and Elementary Properties 4
2.1.1 Definitions and Elementary Properties 5
2.1.2 Notation 5

3 General Purpose Factoring Algorithms **5**
3.1 Pollard (p - 1) Method 5
3.2 Pollard Rho Method 7
3.3 Small Prime Trial Division 11
3.4 Fermat's Factoring Method 11
3.5 Elliptical Curves Method 12
3.6 Multiple Polynomial Quadratic Sieve 13
3.7 Primality Tests 15
3.7.1 Fermat Probable Primes 15

4 Recurrence Relations **15**
4.1 Fibonacci Numbers 16
4.1.1 Fibonacci Factorization Algorithm 16
4.1.2 Results 17
4.2 Lucas Numbers 18
4.2.1 Results 18
4.3 Tribonacci Numbers 18
4.4 Results 18
4.5 $G_n = G_{n-1} + G_{n-3}$ Numbers 19
4.5.1 Results 19

5 Application of Methods **19**

6 Results **22**

7 Platform **24**

8 Future Work **25**

A Tables of Factorizations **26**

B Properties Discovered **26**
B.1 Proofs 26

1 A Brief Thank you

I would like to thank all those people that helped me accomplish this work. I'd first like to thank my thesis committee, Dr. Stanisław Radziszowski and Dr. Peter Anderson, of the Rochester Institute of Technology, and Dr. Pat Arpaia of Saint John Fisher College, for their collective effort in helping me make this paper both intelligible, and hopefully meaningful. I would also like to thank Dr. Dan Cass of Saint John Fisher College, for helping me discover and prove many properties of the Tribonacci numbers.

Finally, I would like to dedicate this thesis to my wife, Mary Jo. Without her ever present love and support, my studies and thesis would be of no more value to me than the most insignificant of matters. Thank you my love.

2 Introduction

Factoring integers has been a popular topic in the field of number theory since shortly after the integers were discovered. Within the past twenty-five years the factorization of large[1] integers has become popular in the field of *algorithmic number theory*, due almost exclusively to the availability of inexpensive, high performance computers. This paper discusses some of the most common factorization algorithms, both classical and modern, specifically in terms of factoring integer sequences defined by second and third order recurrence relations.

Many papers have been published regarding factoring particular types of numbers; among these are: *Tables of Fibonacci and Lucas Factorizations* by John Brillhart, Peter Montgomery and Robert Silverman [8], and *Factorization of the tenth and eleventh Fermat numbers* by Richard Brent [3]. There are also numerous books on the subject of modern factorization, including two that are required reading: *Prime Numbers and Computer Methods for Factorization*, by Hans Riesel [14] and the classic *The Art of Computer Programming (volume 2)* by Knuth [9].

In Section 3 we describe general purpose factoring algorithms, including some elementary complexity analysis of some of the algorithms. The properties and specialized factoring algorithms of the integer sequences considered in this paper are discussed in section 4. Section 5 is a description of how the algorithms and methods discussed are applied, and the computational results of applying the factoring algorithms is covered in section 6. A supplement to this paper contains the actual factorization tables.

We finish our introduction with a few words on definitions, properties and notation used throughout this paper.

2.1 Definitions, Notation and Elementary Properties

It is assumed the reader has a familiarity with *elementary number theory* and *group theory*. A little knowledge of *algebraic number theory* and *field theory (abstract algebra in general)* makes some of the more obtuse algorithms a bit more comprehensible. Suggested reading includes *An Introduction to the Theory of Numbers*, by Hardy and Wright [6], and *Abstract Algebra*, by Herstein [7].

[1] Consider a number large if it has more than 30 digits.

2.1.1 Definitions and Elementary Properties

Theorem 2.1.1 *(Fermat's Theorem) If p is a prime, then $a^{p-1} \equiv 1 \bmod p$, for all $a \not\equiv 0 \bmod p$.*

2.1.2 Notation

$f^n(x)$ the n^{th} functional composition of f.
$gcd(a, b)$ the greatest common divisor of a and b.
G_n - the n^{th} number in the third order recurrence relation defined by $G_n = G_{n-1} + G_{n-3}$.
M_n - the multiplicative group of all primitive residue classes mod n.
p - is always reserved to represent a prime.
S_m an integer sequence periodically recurrent mod m.
T_n the n^{th} Tribonacci number.
U_n the n^{th} Fibonacci number.
V_n the n^{th} Lucas number.
$\mathcal{Z}_n$ the positive integers mod n.
$\mathcal{Z}_+$ the positive integers.
(a/b) - Lengendre's symbol.
$\binom{a}{b}$ binomial coefficient

3 General Purpose Factoring Algorithms

This section is intended to familiarize the reader with general purpose factoring algorithms for numbers without special form.

3.1 Pollard (p - 1) Method

The Pollard $p-1$ method is based on the following two observations. First, if $p-1|Q$, and $\gcd(a,p) = 1$, then $p|a^Q - 1$. This follows directly from Fermat's Theorem[2]. Second, suppose for some factor p of N, that

$$p - 1 = \prod_i q_i^{\alpha_i} \tag{1}$$

where each $q_i^{\alpha_i} \leq B$, B a relatively small bound (i.e., $p-1$ is composed of only small factors).

[2]Since $a^Q - 1 = a^{(p-1)k} - 1 = (a^k)^{p-1} - 1$, for some k, and since p - 1 | Q, by Fermat's theorem $(a^k)^{p-1} \equiv 1 \bmod p$, since $\gcd(a^k, p) = 1$. Therefore $p|a^Q - 1$

If supposition (1) is true, and we let Q be the product all primes and prime powers less than our limit B ($Q = \prod r_i^{\beta_i}$, for all $r_i^{\beta_i} \leq B$, r_i prime and $\beta_i > 0$), then Q is a multiple of $p-1$, and $p = \gcd(a^Q - 1, N)$ is a non-trivial factor of N. If on the other hand $p-1$ contains a factor $f > B$, then in general the algorithm fails.

Algorithm 3.1.1 *(Pollard (p 1) Factoring Algorithm)*

1. *(initialize) Choose a search limit B,*
   choose a seed value a_0 (e.g. $a_0 = 13$),
   let $j = 1$ (counter),
   let Factor Base = all primes and prime powers less than or equal to B ($\{q_i^{\alpha_i} | \ for\ all\ q_i^{\alpha_i} \leq B,\ q\ prime\}$). Assume Factor Base is ordered, and that the largest index is J.

2. $a_j = a_{j-1}^{p_j} \bmod N$, *where p_j is the j^{th} element of Factor Base.*

3. *If* $\gcd(a_j - 1, N) \neq 1$, *then* $\gcd(a_j - 1, N)$ *is a factor of N and we stop.*

4. $j = j + 1$

5. *if $j \leq J$, then Goto 2*

This method can dramatically speed up factorization if the number contains a factor p such that $p-1$ is comprised only of small factors. When this is not the case however, the algorithm is no better than trial division, so the method is used with a relatively small bound on the factors of $p-1$.

There is an extension to the $p-1$ algorithm that can be used if no factors are found within the search limit B. Suppose, as in (1) that

$$p - 1 = \prod_{i=1}^{j} q_i^{\alpha_i} = Q \prod_{i=1}^{j-1} q_i^{\alpha_i},$$

where Q is the only prime factor of $p-1$ that exceeds B. Now let $b = \prod q_i^{\beta_i} \bmod N$, *for all* $q_i^{\beta_i} \leq B$ (i.e., the final value of the calculation of step 2 of the $p-1$ algorithm above), and choose a new search limit B_2 such that $B < B_2$. Define the set R as the set of all primes between the largest prime less than B and B_2 ($\{r_i | r_i \geq$ (largest prime $< B$), and $r_i < B_2\}$). Now recursively find the value

$$b^{r_{i+1}} = b^{r_i} b^{(r_{i+1} - r_i)} mod N, \tag{2}$$

if $\gcd(b^{r_{i+1}} - 1, N) > 1$, we've found a factor.

Algorithm 3.1.2 *(Pollard (p 1) Factoring Algorithm Phase 2)*

1. *(initialize)Choose a new search limit* B_2,
   let b equal the final computation of a in step 2 of algorithm 3.1.1,
   define $\{r_i | r_i \geq$ *largest prime* $< B$, *and* $r_i < B_2\}$*; assume ordered, with largest index I.*

2. *calculate* $b^{r_{i+1}} = b^{r_i} b^{(r_{i+1}-r_i)} \bmod N$

3. *If* $\gcd(b^{r_{i+1}} - 1, N) \neq 1$, *then* $\gcd(b^{r_{i+1}} - 1, N)$ *is a factor of N and we stop.*

4. $i = i + 1$

5. *if* $i \leq I$, *then Goto 2*

The second phase of this algorithm was not used, but is presented here for completeness.

3.2 Pollard Rho Method

Pollard's Rho method is a Monte Carlo algorithm that usually finds a factor, not necessarily prime, of an integer N in $O(N^{1/4})$ arithmetic operations.

The ideas behind Pollard's Rho method can be described in gross terms[3] as follows:

1. Given a composite number N, and p, an unknown factor of N, construct a sequence S that is periodic mod p.

2. Determine the period of S.

3. Determine p using the above sequence and its period.

The first thing we must do is find a way to *easily* construct an integer sequence that is periodic mod p. Consider the sequence S_m (m a positive integer), defined by,

$$S_m = \{x_i | x_i = f(x_{i-1}) \bmod m, \forall i \in \mathcal{Z}_+, 0 < x_0 < m, f \text{ a polynomial}\}. \quad (3)$$

The sequence S_m can be constructed by taking the successive functional composition of an initial value (x_0) for some polynomial f (throughout the

[3]As you will soon see this is a grossly oversimplified description, but it will do as a starting point.

remainder of this section any function f should be assumed to be a polynomial, unless otherwise stated), then calculate that value modulo some prime p. Constructing the sequence S_m is computationally inexpensive, and showing that S_m is periodic is not difficult.

Lemma 3.2.1 *S_m is periodic.*

Proof. Applying the *Pigeon Hole principle* to the first $m+1$ elements of S_m, some value $f^i(x_0)$ must be repeated. Let i be the index of the first occurrence of $f^i(x_0)$, and let j be the index of the next occurrence of $f^i(x_0)$ (i.e., $f^i(x_0) = f^j(x_0)$). Label $f^i(x_0) = a$. Then for all $k \geq 0, f^{i+k}(x_0) = f^k(a) = f^{j+k}(x_0)$ (the fact that f is a polynomial is necessary, as you can see). So S_m is periodic from the index i on.

Lemma 3.2.2 *If S_m is periodic and $x_i, x_j \in S_m, j > i$, such that, $x_i = x_j$, then S_m is periodic starting at x_i.*

Proof. Proved as part of the proof of Lemma 3.2.1

Now that we can construct S_m with no great difficulty, suppose we wish to find its period. Obviously a direct search comparing arbitrarily long subsequences will work, but is out of the question for sequences with sufficiently large period-lengths. Fortunately there exist algorithms to detect the period that are efficient. Specifically, *Floyd's cycle-finding algorithm*[4] detects the period of a periodic sequence $\{x_i\} \bmod m$ by testing if $x_{2i} \equiv x_i \bmod m$.

Theorem 3.2.1 *(Floyd's Cycle Finding Theorem) The period of a periodic sequence S_m can be detected by comparing $x_{2i} \equiv x_i \bmod m$.*

Proof. Let a be the aperiodic length and l be the period-length of S_m. Let $j \geq i$, $i \geq a$, then $x_j \equiv x_i \bmod m$ implies that $j - i = kl$ for some $k \geq 0$ (i.e., $j - i$ is a multiple of the period). So, if $x_{2i} \equiv x_i \bmod m$, then i is a multiple of the period, and the sequence is periodic from x_{2i} onwards. Conversely, if $x_j \equiv x_i \bmod m$ is true for $j = i + kl$, for $k > 0$, $\forall\ i > a$. Eventually there will be an i such that $j = 2i$ (specifically when $kl = i$)[5].

Algorithm 3.2.1 *(Floyd's Cycle-Finding Algorithm)*

[4]Richard Brent [2] has found a more efficient algorithm than Floyd's, and it is generally used. We use Floyd's algorithm in this explanation because of its simplicity.

[5]For a more in depth treatment of this algorithm see [2, 9, 14]

1. *Given a period sequence S_m as defined in (3).*

2. *(initialize)* $x = x_1, y = f(x_1)$

3. *Repeat* $\{x = f(x), y = f(f(y))\}$ *Until* $x = y$.

Once Floyd's algorithm terminates, the period can be found by comparing the terminating value of x, which we will call x_i with $x_{i+1}, x_{i+2}, \ldots$ until $x_{i+l} = x_i$, at which point we know l is the period-length.

Pollard applied Floyd's algorithm to simple polynomials $f(x) \bmod N$ (specifically quadratics of the form, $f(x) = x^2 + a, a \neq 0$ *or* -2)[6], and replaced the termination test $x = y$ with *if* $\gcd(|x - y|, N) > 1$, *then stop.* Pollard also showed that there is no need to compute the greatest common divisor on every iteration [12], but that we can instead calculate

$$Q_n = \prod_{i=1}^{n} (x_{2i} - x_i),$$

and apply Euclid's algorithm only occasionally. The chances that the algorithm will fail because $Q_n = 0$ does increase, so we shouldn't perform Euclid's algorithm too infrequently.

Algorithm 3.2.2 *(Pollard's Rho Factoring Algorithm)*

1. *(initialize) Choose a quadratic* $f(x) = x^2 + a, a \neq 0$ *or* -2,
   choose $x_1 \in \{1, 2, \ldots, N-1\}$,
   choose a frequency c *with which to apply Euclid's algorithm,*
   choose a search limit B,
   let $Q_0 = 1$,
   let $i = 1$.

2. $x = x_1, y = f(x_1)$

3. $Q_i = (y - x)Q_{i-1} \bmod N$

4. *if* $i \equiv 0 \bmod c$ *and* $\gcd(Q_i, N) > 1$, *then stop*

5. $x = f(x), y = f(f(y))$.

[6] $a = 0$ should be avoided for obvious reasons; we quote Knuth [9, p. 371]: '$a = -2$ should also be avoided, since the recurrence $x_{m+1} = x_m^2 - 2$ has solutions of the form $x_m = r^{2^m} + r^{-2^m}$. Other values of a do not seem to lead to simple relationships mod p.'

6. $i = i + 1$.

7. *if* $i \leq B$ *Goto 4, else stop.*

Although Pollard's Rho algorithm is as beautifully simple as Floyd's cycle finding algorithm, why should we expect it to produce the unknown factor p? Consider the sequence S_N generated by the function $f(x) = (x^2+a) \bmod N$. According to Lemma 3.2.1, S_N is periodic (although the period-length could be large). Now consider the sequence S_p generated by the function $f(x)$ given in S_N, where p is the smallest prime factor of N (i.e., $S_p = S_N \bmod p$). If $x_{2i} \equiv x_i \bmod N$, then it follows that $x_{2i} \equiv x_i \bmod p$, so we can think of the test $\gcd(|x-y|, N) > 1$ on S_N as simply being a redundant form of the same test on S_p, which will terminate in at most $i \leq p$ iterations.

The worst case for Pollard's algorithm then is p iterations. Brent [2] and Knuth [9, p. 367-369] have shown that the expected value[7] of i is approximately $(\pi/2)^{5/2}p^{1/2}$. So the number of iterations of Pollard's algorithm needed to find the factor p is on the order of $O(N^{1/4})$.

An intuitive (and necessarily less rigorous) algebraic argument showing why a factor p of N (where N is sufficiently large to apply the probabilistic arguments that follow) is likely to be found in $O(N^{1/4})$ iterations of Pollard's Rho method is given by Riesel [14, p. 182-183]:

$$\begin{aligned} y_i &= x_{2i} - x_i = x_{2i-1}^2 + a - x_{i-1}^2 - a \\ &= x_{2i-1}^2 - x_{i-1}^2 = (x_{2i-1} + x_{i-1})(x_{2i-1} - x_{i-1}) \\ &= (x_{2i-1} + x_{i-1})(x_{2i-2} + x_{i-2})(x_{2i-2} - x_{i-2}) \\ &\cdots \\ &= (x_{2i-1} + x_{i-1})(x_{2i-2} + x_{i-2})...(x_i + x_0)(x_i - x_0) \end{aligned}$$

So there are $i+1$ algebraic factors in y_i. How many iterations do we need to ensure that all primes less than or equal to p are included in the y_i's? Since the number of prime factors $\leq G$ of a given number N is approximately $\ln(\ln(G))$, see [14, p. 161], executing n iterations of Floyd's algorithm we have $Q_n = \prod_{i=1}^{n} y_i$, which should contain

$$\ln(\ln(G)) \sum_{i=1}^{n} (i+1) \approx 0.5n^2 \ln(\ln(G))$$

[7]The statistical distribution of the total number of prime factors of a number N is approximately normal [9, p. 368].

prime factors $\leq G$. We also know that $\pi(p) \approx p/\ln(p)$, which gives us

$$p/\ln(p) \approx 0.5n^2 \ln(\ln(p)),$$

$$n \approx p^{1/2}.$$

So the number of iterations of Pollard's algorithm is on the order of $O(p^{1/2}) = O(N^{1/4})$.

The Pollard Rho algorithm can be further modified by limiting the number of iterations of the main loop. The more iterations, the higher the probability of success. Choosing the number of iterations to be $\approx 1.18p^{1/2}$ yields approximately a 50% success rate, see [14, p. 179-180].

3.3 Small Prime Trial Division

Trial division with the first 100,000 primes eliminates small prime factors that may appear in factors found using algorithms such as Pollard Rho.

Since the number of primes we are using is relatively small, the trial division algorithm can be optimized by maintaining a list of primes cached in memory.

3.4 Fermat's Factoring Method

The idea behind Fermat's method is to express a composite number N as the difference of two squares, say, $x^2 - y^2$. In other words, if we can express

$$N = x^2 - y^2 = (x - y)(x + y), \tag{4}$$

in a non-trivial manner, then we have found two non-trivial factors of N. In order to arrive at an efficient algorithm we rewrite (4) as:

$$y^2 = x^2 - N. \tag{5}$$

From this we see that $x > \sqrt{N}$, so we Initially set $x = \lfloor \sqrt{N} + 1 \rfloor$; if $x^2 - N$ is a perfect square, a factor has been found; otherwise increment x and try again. Notice that $(x+1)^2 - N = x^2 + 2x + 1 - N = y^2 + (2x+1)$. So, when we need to increment x, we can simply add $2x + 1$ instead of the squaring x and subtracting N. This observation makes the algorithm more efficient.

Algorithm 3.4.1 *(Fermat's Factoring Algorithm)*

1. *(initialize)* $m = \lfloor \sqrt{N} + 1 \rfloor$,
   choose a search limit B.

2. *Calculate* $z = m^2 - N$

3. *If z is a perfect square, then stop (N is a difference of squares)*

4. $z = z + (2m + 1)$

5. $m = m + 1$

6. *If* $m > B$, *then stop.*

7. *Goto 3*

Fermat's algorithm is only efficient when N contains two nearly equal factors. By reasonably limiting the search, we can occasionally save a lot of computation time.

3.5 Elliptical Curves Method

H. W. Lenstra discovered the Elliptic Curve Method (ECM) of integer factorization. Lenstra applied Pollard's $p - 1$ method over a group other than M_n (the multiplicative group of all primitive residue classes $\bmod n$). Specifically, Lenstra used the group of rational points on an elliptic curve[8].

There are several forms of elliptic curves; the most commonly used form is due to Peter Montgomery,

$$By^2 = x^3 + Ax^2 + x, \quad with\ B(A^2 - 4) \neq 0.$$

By varying A and B we can generate several groups over which the algorithm can be used; for details on why this form is desirable see [3, 14].

The algorithm from this point on is almost identical to the Pollard $P-1$ algorithm. Choose A and B (there are prescribed ways makes these choices) and a rational starting point $P_1 = (x_1, y_1)$. Now iteratively calculate P_i by

$$P_{i+1} = p_i \circ P_i \bmod N,$$

where p_i is the i^{th} prime and $\circ$ is the group composition operation. Periodically check if $1 < \gcd(x_{i+1}, N) < N$, until a factor is found, or a pre-defined search limit is reached. If a factor isn't found, change A, B, and P_1, and

[8]Jacobi discovered the remarkable fact that the set of rational points on an elliptic curve form a group under an operation of composition that generates a rational point from two others.

start again. After changing curves a number of times, a factor is usually found.

The ECM algorithm is very effective at finding factors in the 15 to 40 digit range.

3.6 Multiple Polynomial Quadratic Sieve

The Quadratic Sieve algorithm developed by C. Pomerance [13], depends on finding non-trivial solutions to the Legendre congruence

$$A^2 \equiv B^2 \bmod N. \tag{6}$$

Another, more useful way to express this congruence is

$$A^2 - B^2 \equiv (A - B)(A + B) \equiv 0 \bmod N. \tag{7}$$

If $A \not\equiv B \bmod N$ and $A \not\equiv -B \bmod N$, then $\gcd(A + B, N)$ and $\gcd(A - B, N)$ are proper factors of N (notice that this is essentially the same technique used in Fermat's method. Using quadratic congruences is the basis for nearly all modern factoring algorithms (Quadratic Sieve, Multiple Polynomial Quadratic Sieve, Number Field Sieve, Continued Fractions algorithm) of sufficient power to factor numbers without special form of greater than 30 digits). Pomerance's original Quadratic Sieve (single polynomial) used the following quadratic equation to generate quadratic residues,

$$Q(x) = (x + floor(\sqrt{N}))^2 - N \equiv H^2 \bmod N. \tag{8}$$

Pomerance recognized that using this polynomial to generate a set of quadratic residues of N, provides a rather nice consequence; namely, if $p|Q(x)$, then $p|Q(x + kp)$ for all $k \in \mathcal{Z}$. To see this substitute $(x + kp)$ for x in $Q(x)$, and notice that,

$$Q(x + kp) = Q(x) + (kp)^2 + 2kp(x + floor(\sqrt{N})) \equiv 0 \bmod p. \tag{9}$$

So, with only a single solution x to equation (8) such that $p|Q(x)$, other values x_i can be found such that $p|Q(x_i)$ by sieving on x (i.e., $x_i = x \pm kp$ where $k = 0, \pm 1, \pm 2, \ldots$).

Algorithm 3.6.1 *(Quadratic Sieve)*[9]

1. *Select a set of primes called the factor base, FB, such that $FB = \{p_i | (N/p_i) = 1, i = 1, 2, ..., B\} \bigcup \{p_0 = -1\}$ (where (N/p_i) is the Legendre symbol), for an appropriate bound B (choosing an appropriate bound B is non-trivial, and is not described here). The goal here is to find a set of primes such that $Q(x)$ is p_B smooth, i.e., $Q(x)$ can be factored completely over the chosen factor base.*

2. *Find the roots r_1 and r_2 of the polynomial $Q(x) \equiv 0 \bmod p_i$, for all $p_i \in FB$.*

3. *Initialize a sieve array $S[x] = 0$ over some interval [-M, M] for some appropriate value of M.*

4. *$\forall p_i \in FB$, add $\ln(p_i)$ to the $(r_j + kp_i)$ location of S (i.e., $S[r_j + kp_i] = S[r_j + kp_i] + \ln(p_i)$), where $j \in \{1, 2\}, k \in \mathcal{Z}, r_j + kp_i \in [-M, M]$.*

5. *Since the value of $Q(x)$ over [-M, M] is approximately $M\sqrt{N}$, look for sieve locations where $S[x] \approx \ln N/2 + \ln M$. It is precisely at these locations that $Q(x)$ may have fully factored residues, and we perform trial division to factor $Q(x)$,*

$$Q(x) = \prod_{i=0}^{B} p_i^{\alpha_i},\ p_i \in FB. \tag{10}$$

6. *Let $\omega_j = (\alpha_1, \alpha_2, ..., \alpha_B)$ and $H_j^2 = Q(x)$ as defined in equation (10) .*

7. *Generate $B + 1$ of the factorizations in (6) and create an augmented matrix M, where $R_j = [\omega_j | H_j^2]$ is the j^{th} row, and reduce it via Gaussian elimination over Z_2. If a linear dependency exists in M, then the product of the vectors in that dependency forms a square, and constructing a quadratic congruence (6) is straight forward.*

The computational effort required to factor a number with the Quadratic Sieve algorithm is a function of the size of the number being factored. This is different from most classical factorization methods where the computational effort is a function of the second largest factor. Since the computational cost is dependent on a known value, heuristics can be devised to estimate the

[9]The description of this algorithm is based largely on a paper by R. Silverman [15].

run time of the algorithm. In so doing it doesn't take long to discover that as N grows, so do M and the factor base (as N increases, $Q(x)$ increases and as $Q(x)$ increases so does the value p_B such that $Q(x)$ is p_B smooth).

This problem is a significant deficiency in the algorithm when applied to sufficiently large numbers. Peter Montgomery [15] came up with the idea to use multiple polynomials of the form

$$Q(x) = Ax^2 + Bx + C, \; with \; B^2 - 4AC = N, \tag{11}$$

to generate the desired quadratic residues (note that $B^2 - 4AC = N$ is not arbitrary and is indeed necessary to ensure we can generate quadratic residues, see [15]). Using multiple polynomials means that the size of the sieve interval can be kept small, resulting is residues that are in general easier to factor. The use of multiple polynomials also lends itself directly to parallel implementations.

There are many variants of MPQS which optimize different parts of the algorithm. For a more complete treatment of these variations see [11, 4, 5, 10]. The MPQS was used to solve the RSA-129 challenge.

3.7 Primality Tests

3.7.1 Fermat Probable Primes

To test the primality of discovered factors (specifically large factors) a version of the Fermat probable prime test was used. Each probable prime was tested using 100 different bases, so the probability that the probable primes discovered are composite is unlikely (but of course, not impossible). For details on Fermat probable prime algorithms, see [9, 1].

4 Recurrence Relations

Let a_1, a_2, and a_3 be integers. If $a_3 \neq 0$, then

$$r_n = a_1 r_{n-1} + a_2 r_{n-2} + a_3 r_{n-3}, \forall n \geq 3,$$

along with initial values for r_0, r_1 and r_2, is a third order recurrence relation. To express a second order recurrence relation, we let $a_3 = 0$ and $a_2 \neq 0$,

$$r_n = a_1 r_{n-1} + a_2 r_{n-2}, \;\; n \geq 2.$$

As mentioned earlier, the recurrence relations considered in this paper are the second and third order recurrence relations which generate the Fibonacci, Lucas and Tribonacci numbers, as well as the third order recurrence relation defined by $r_n = r_{n-1} + r_{n-3}$.

4.1 Fibonacci Numbers

The Fibonacci numbers are defined by the second order recurrence relation

$$U_n = U_{n-1} + U_{n-2}, \quad n \geq 2, \tag{12}$$

where $U_0 = 0$, and $U_1 = 1$.

The Fibonacci numbers are perhaps the most well known of all integer sequences. They make regular appearances in nature (pine cones and sea shells for example), art, architecture, and a host of other fields. There is also voluminous amounts of information regarding the Fibonacci numbers available that can be used to verify or refute the results in this paper[10].

4.1.1 Fibonacci Factorization Algorithm

Factoring Fibonacci numbers can be simplified by the fact that the greatest common divisor of two Fibonacci numbers is the Fibonacci number whose index is the greatest common divisor of the indices of the Fibonacci numbers in question,

$$\gcd(U_m, U_n) = U_{\gcd(m,n)}. \tag{13}$$

From this simple identity we can prove the following theorem.

Theorem 4.1.1 *$U_n = C \prod_{i=1}^{k} U_{p_i^{e_i}}$, where $n = \prod_{i=1}^{k} p_i^{e_i}$.*

Proof. From equation (13) we know that for each of the factors $p_i^{e_i}$ of n,

$$\gcd(U_n, U_{p_i^{e_i}}) = U_{p_i^{e_i}},$$

and,

$$\gcd(U_{p_i^{e_i}}, U_{p_j^{e_j}}) = 1.$$

Since all of the $U_{p_i^{e_i}}$ are pairwise relatively prime and each divides U_n,

$$U_n = C \prod_{i=1}^{k} U_{p_i^{e_i}},$$

[10] See Vajda [16] and Vorobyov [17]

for some constant C.

We can now construct an algorithm to factor Fibonacci numbers with composite indices.

Algorithm 4.1.1 *(Fibonacci Factoring Algorithm 1)*

1. *Given U_n, where n is composite, factor n into its prime factorization $p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$.*
2. *Factor each $U_{p_i^{e_i}}$, use the factorization if we already know it.*
3. *Factor C.*

Another property of Fibonacci numbers that can be exploited when factoring is the identity,

$$U_{2n} = U_n V_n, \tag{14}$$

where V_n is the n^{th} Lucas number. Equation (14) leads to an efficient factoring algorithm for Fibonacci numbers with even indices.

Algorithm 4.1.2 *(Fibonacci Factoring Algorithm 2)*

1. *Given U_{2n} factor U_n and V_n, or copy the factorizations if we already know them.*

The real difficulty in factoring Fibonacci numbers is when the index is prime, and then general purpose factoring algorithms are our only recourse.

For an in depth treatment of the multiplicative structure of the Fibonacci numbers, see [8].

4.1.2 Results

We successfully factored the first 382 Fibonacci numbers (plus a few others of higher index), and verified these factorizations against the work of Brillhart, Montgomery and Silverman [8] (this work contained the complete factorization of all Fibonacci numbers up to the 388^{th}). We also factored completely the 389^{th} Fibonacci number, the first number not completely factored in Brillhart, Montgomery and Silverman's paper. I have since contacted Peter Montgomery and received an updated table of Fibonacci factorizations, and not only has the 389^{th} Fibonacci number been factored since the publication of the original paper; but the first Fibonacci number not completely factored is U_{703}.

4.2 Lucas Numbers

The Lucas numbers are defined by the second order recurrence relation,

$$V_n = V_{n-1} + V_{n-2}, \quad n \geq 2, \tag{15}$$

where $V_0 = 2$, $V_1 = 1$.

The Lucas numbers are a close cousin to the Fibonacci numbers. There are two reasons for including them: 1) they can be used to aid in the factorization of Fibonacci numbers (see above) and 2) they have well known multiplicative properties (and existing factorization tables), providing a means to verify the implementation of the general purpose factoring algorithms.

4.2.1 Results

While there exists a useful multiplicative structure to the Lucas numbers (again, see [8]), these properties were not used as an aid to factorization. Instead only the general purpose factoring algorithms at our disposal were used, so that a verification of our implementation could be assessed. The first 185 Lucas numbers were successfully factored in less than an hour of real time (half of these numbers contain 20-40 digits).

4.3 Tribonacci Numbers

The Tribonacci numbers are defined by the third order recurrence relation

$$T_n = T_{n-1} + T_{n-2} + T_{n-3}, \quad n \geq 3, \tag{16}$$

where $T_0 = 0$, and $T_1 = T_2 = 1$.

This is perhaps the most enigmatic sequence of integers we considered. There is a definite multiplicative structure, but it is convoluted and appears to be of no use as an aid in the factorization of numbers higher in the sequence.

4.4 Results

We successfully factored the first 311 Tribonacci numbers (as well as a few others of higher index). To the best of our knowledge this is the most extensive factorization of the Tribonacci numbers; we could locate no references to work that had attempted to factor the Tribonacci numbers.

4.5 $G_n = G_{n-1} + G_{n-3}$ Numbers

The final sequence considered is the one generated by the third order recurrence relation

$$G_n = G_{n-1} + G_{n-3}, \quad n \geq 3, \tag{17}$$

where $G_0 = 0$, and $G_1 = G_2 = 1$. From this point on we will call this the G sequence; if a name exists for this sequence already, no slight is intended.

Only the general purpose factoring algorithms were applied to these numbers.

4.5.1 Results

We successfully factored the first 322 numbers in this sequence.

5 Application of Methods

If there is a *best* strategy for factoring arbitrarily large composite integers without special form, it is not yet known. However, most researchers agree that the following set of guidelines work reasonably well, see [8, 14]:

1. Perform trial division up to some small limit.
2. Make sure N passes a compositeness test (i.e., it isn't a probable prime).
3. Apply Pollard's Rho algorithm, searching for factors in the range of 8 to 15 digits.
4. Now is where you hope for a little luck and try algorithms that occasionally find large factors, but in general either fail, or are too time consuming to be useful in the general case (for-example: Pollard's $P-1$ and Fermat's algorithms). These algorithms should be run with relatively small bounds, since they are unlikely to find a factor in the general case.
5. Use ECM to search for factors that are near or beyond the upper limit of the particular Pollard Rho implementation you are using (i.e., find the efficiency crossover point of the Pollard Rho and Elliptic Curves Method for the specific system being used).

6. When all else fails use the 'big guns: MPQS or the Number Field Sieve[11].

The strategy used to factor the integers considered in this paper follows the recipe given above, and was as follows:

1. Given a number N to factor, first make sure N is not a Fermat probable prime (repeat this test on the quotient, any time a factor is found in any step). The probability that number is actually composite after passing the Fermat probable prime test is $(1/4)^{100}$ (see [9, p. 379]).

2. If we are factoring Fibonacci numbers, then we need to use the special case Fibonacci factoring algorithms now. If we waited until some of the factors are found by other algorithms (say during trial division), then we have to sort out the duplicates, since the special case algorithms involve copying factorization of previous Fibonacci numbers.

3. Now perform trial division using the first 100,000 primes (yes, we used the first 100,000 primes, not all primes under 100,000). The limit on the trial division ($p_{100,000} = 1299709$) should eliminate for all practical purposes the need to factor the factors found by the remaining algorithms.

4. Apply Fermat's algorithm with a bound of 100,000 iterations. This algorithm is extremely fast with such a small bound, and should find factors that are within a range of 100,000 of the square root of the number we are factoring.

5. Pollard's Rho algorithm is now run, with an upper bound on the number of iterations at 100,000, so we should expect to find factors in the 6 to 11 digit range. The range of digits we should expect comes from the fact that trial division eliminates factors less than 6 digits, and the maximum number of iterations is a hidden representation of the square root of the largest prime factor the algorithm can detect (refer back to the closing remarks of section 3.2), i.e., $100,000^2$ has 11 digits.

[11]The Number Field Sieve is the latest, and most powerful modern factorization algorithm. This is the algorithm used to factor the RSA-130 number. This algorithm relies heavily on *algebraic number theory* and is beyond the scope of this paper.

6. If we found a factor using Pollard's Rho algorithm in step 5, then go back and continue from step 4. If we didn't find a factor, then either change the polynomial and return to step 5, or continue if we have already changed polynomials more than 3 times (the choice to use 3 polynomials was arbitrary).

7. Try Pollard's $P - 1$ algorithm, just in case.

8. Onto the Elliptic Curve Method. An existing implementation of this algorithm was used[12] to search for factors in the 10 to 34 digit range. The reason for this range is simple. At the lower end, the Pollard Rho algorithm is in general going to find factors that are less than 11 digits. The upper end is a limitation of implementation.

9. Now pull out the big gun, namely, MPQS.

It may seem that 100,000 was chosen arbitrarily as the bound limit for most of the algorithms. There is a small amount of truth in that, but the bounds were actually chosen to ensure that each subsequent algorithm was not duplicating a search for factors of the previous algorithms. In the case of Fermat's algorithm, we choose this bound only to preserve this coincidental symmetry.

[12]The LiDIA C++ library of computational number theory routines

6 Results

Below we present the factors found using each algorithm (except trial division) on all the numbers of 50 or fewer digits in each sequence[13].

Sequence	Rho	ECM	P-1	Fermat	MPQS	Prob Prime	Total
Fibonacci	42	12	0	0	0	168	222
Fibonacci*	90	24	0	2	0	175	291
Lucas	90	36	0	0	0	193	319
Tribonacci	61	23	0	0	0	141	225
G	101	39	0	2	0	241	383
Total	384	134	0	4	0	918	1440

Table 1: factors found by each algorithm.

As the numbers being factored increased in size, ECM and MPQS became more dominant, which is the expected outcome. This is not reflected in the table above since the crossover doesn't occur until around 60 digits.

In previous trials MPQS started extracting factors around the 40 digit range, however, the data above reflects a rather selfish ECM algorithm (which was contrived, in order to avoid problems with the implementation of some of the software routines not implemented directly by the author).

[13]In the following tables, Fibonacci* means the special case Fibonacci factoring algorithms were not used

Now we present the same data normalized by the number of factors found,

Sequence	**Rho**	**ECM**	**Prob Prime**	**Total**
Fibonacci	0.19	0.05	0.76	222
Fibonacci*	0.31	0.08	0.60	291
Lucas	0.28	0.11	0.61	319
Tribonacci	0.27	0.10	0.63	225
G	0.26	0.10	0.63	383
Total	0.27	0.09	0.64	1440

Table 2: factors found by algorithm normalized by total factors found.

At first glance the number of factors detected using the Fermat probable prime test may seem suspiciously high, however, once you consider that the last factor (when N is sufficiently large) will almost always be detected this way, the numbers becomes much more reasonable.

Approximately 30% of all factors (excluding those found using trial division) were extracted using Pollard's Rho algorithm, and 10% were extracted using ECM. The average size of a factor was 8 digits for Pollard Rho and 15 digits for ECM (this was uniform across all four sequences), which is approximately the midpoint of the range of each algorithm respectively.

Sequence	Avg. Size of Prob. Primes
Fibonacci	15
Lucas	13
Tribonacci	18
G	18

Table 3: Avg. size of Probable Primes (50 digits or less)

The table above gives an indication of the average size of the largest factor of the numbers factored in each sequence. It is interesting to note, that the average size of the largest factor in the third order recurrence relations, seem to be larger than for those of the second order recurrence relations. This result supports an observation we made about the factors in the Tribonacci sequence, namely, if you examine the table of factorizations of the Tribonacci's, it appears that there are inordinately many factors that are larger than expected, assuming a Gaussian distribution of factors. It seems that this difference is not caused by the different growth rates of the second and third order recurrence relations either. We tested this hypothesis by normalizing the data in table 3. We did this by dividing by the average size of the numbers considered in each sequence, in the range of 1-50 digits. This result could be explained by the small sample space used in this experiment, but the author believes it is worth investigating given the preliminary evidence.

7 Platform

The platform used to develop the software and perform the experiments is:

- SUN Ultra Sparc 1 (64MB memory and 1.2G hard drive),
- C/C++ programming languages,
- GNU Multiple Precision Integer library (-lgmp).
- LiDIA C++ Algorithmic Number Theory library (-lLiDIA)

- GNU C++ compiler,
- SUN Sparc Works C compiler

The algorithms implemented specifically for the present paper are:

- Pollard $P-1$,
- Pollard Rho,
- Small Prime Trial Division,
- Fermat's Factoring Method,
- Special case Fibonacci factoring algorithms,
- Multiple Polynomial Quadratic Sieve (we experimented with publicly available implementations, as well as our own; the former proved to be more efficient).

The algorithms imported from the LiDIA library are:

- Elliptical Curves Method
- Multiple Polynomial Quadratic Sieve.

The algorithms imported from the Gnu Multiple Precision Integer library are:

- Fermat's Probable Prime Test
- Euclid's Extended Greatest Common Divisor Algorithm.

8 Future Work

There are many questions I have that are left unanswered by this research:

1. The larger Tribonacci numbers seem to have a large number of large prime factors. While this could be explained by random chance, it is rather perplexing.
2. Investigate and implement the Number Field Sieve factoring algorithm.

A Tables of Factorizations

See the supplement that accompanies this paper for the actual factorizations.

B Properties Discovered

During the course of analyzing the factorization tables of the Tribonacci numbers, several interesting patterns were noticed. We spent a considerable amount of time looking for properties that would aid in the factorization of Tribonacci numbers (in short, we were searching for a multiplicative structure like the one in the Fibonacci's), but this effort to date has been of no help in this regard[14].

We present a list of properties of Tribonacci numbers with proofs:

$$\begin{aligned}
T_{n+k} &= T_{n-2}T_k + T_{n-1}(T_{k-1} + T_k) + T_nT_{k+1}, \\
T_{n+k} &= T_{n+1}T_k + T_{k+1}T_n - T_nT_k + T_{n-1}T_{k-1}, \\
T_{2n} &= 2T_nT_{n+1} - T_n^2 + T_{n-1}^2, \\
T_{2n} &= T_{n+1}^2 - (T_{n+1} - T_n)^2 + T_{n-1}^2, \\
T_{2n-1} &= 2T_{n-1}T_{n-2} + T_n^2 + T_{n-1}^2, \\
T_n &= (1/2^k)\sum_{j=0}^{k} \binom{k}{j} T_{n+k-4j}, \\
\sum_{i=0}^{n} T_i &= (T_n + T_{n+2} - 1)/2.
\end{aligned}$$

B.1 Proofs

Lemma B.1.1 $T_{n+k} = T_{n-2}T_k + T_{n-1}(T_{k-1} + T_k) + T_nT_{k+1}$.

Proof (by Dr. Dan Cass). Let M be the matrix generating a shift of a Tribonacci sequence, so that when M left-multiplies a column vector $(a, b, c)^t$ the result should be $(b, c, a + b + c)^t$. Then,

$$M = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

[14] I would like to provide special thanks to Dr. Dan Cass and Dr. Peter Anderson for their contributions in helping discover these properties.

Now let T_n be the n^{th} Tribonacci number; one can then show via induction that using the Tribonacci recurrence that we have the following expression for the matrix power M^k:

$$M^k = \begin{bmatrix} T_{k-2} & (T_{k-3} + T_{k-2}) & T_{k-1} \\ T_{k-1} & (T_{k-2} + T_{k-1}) & T_k \\ T_k & (T_{k-1} + T_k) & T_{k+1} \end{bmatrix}$$

Since M generates a shift on T_n itself we have

$$M^k \begin{bmatrix} T_{n-2} \\ T_{n-1} \\ T_n \end{bmatrix} = \begin{bmatrix} T_{n-2+k} \\ T_{n-1+k} \\ T_{n+k} \end{bmatrix}$$

So T_{n+k} is obtained by using row 3 of M^k and the column vector on the left of the above:

$$T_{n+k} = T_{n-2}T_k + T_{n-1}(T_{k-1} + T_k) + T_nT_{k+1} \tag{18}$$

Lemma B.1.2 $T_{n+k} = T_{n+1}T_k + T_{k+1}T_n - T_nT_k + T_{n-1}T_{k-1}$.

Proof. From lemma B.1.1 we know that

$$T_{n+k} = T_{n-2}T_k + T_{n-1}(T_{k-1} + T_k) + T_nT_{k+1},$$

now replace T_{k+1} with $T_k + T_{k-1} + T_{k-2}$,

$$\begin{aligned} T_{n+k} &= T_{n-2}T_k + T_{n-1}(T_{k-1} + T_k) + T_n(T_k + T_{k-1} + T_{k-2}), \\ &= T_k(T_n + T_{n-1} + T_{n-2}) + T_{n-1}T_{k-1} + T_n(T_{k-1} + T_{k-2}) \\ &= T_kT_{n+1} + T_{n-1}T_{k-1} + T_n(T_{k+1} - T_k) \\ &= T_{n+1}T_k + T_{k+1}T_n - T_nT_k + T_{n-1}T_{k-1} \end{aligned}$$

Lemma B.1.3 $T_{2n} = 2T_nT_{n+1} - T_n^2 + T_{n-1}^2$.

Proof. Express T_{2n} as T_{n+n} and apply lemma B.1.1:

$$\begin{aligned} T_{n+n} &= T_{n-2}T_n + T_{n-1}(T_{n-1} + T_n) + T_nT_{n+1}, \\ &= (T_{n+1} - T_n - T_{n-1})T_n + T_{n-1}^2 + T_nT_{n-1} + T_nT_{n+1}, \\ &= 2T_{n+1}T_n - T_n^2 - T_{n-1}T_n + T_{n-1}^2 + T_nT_{n-1}, \\ &= 2T_{n+1}T_n - T_n^2 + T_{n-1}^2. \end{aligned}$$

Lemma B.1.4 $T_{2n} = T_{n+1}^2 - (T_{n+1} - T_n)^2 + T_{n-1}^2$.

Proof. Express T_{2n} as T_{n+n} and apply lemma B.1.2 to get the result.

Lemma B.1.5 $T_{2n-1} = 2T_{n-1}T_{n-2} + T_n^2 + T_{n-1}^2$.

Proof. Express T_{2n-1} as $T_{n+(n-1)}$ and apply lemma B.1.1:

$$\begin{aligned} T_{n+(n-1)} &= T_{n-2}T_{n-1} + T_{n-1}(T_{n-2} + T_{n-1}) + T_nT_n, \\ &= 2T_{n-2}T_{n-1} + T_{n-1}^2 + T_n^2. \end{aligned}$$

Lemma B.1.6 $\sum_{i=0}^{n} T_i = (T_n + T_{n+2} - 1)/2,\ n \geq 3$.

Proof. The reader can easily verify this is true for $n = 3$. Assume

$$\sum_{i=0}^{n} T_i = (T_n + T_{n+2} - 1)/2, \text{ for some } n \geq 3.$$

Now look at the sum over the first $n+1$ elements,

$$\begin{aligned} \sum_{i=0}^{n+1} T_i &= T_{n+1} + \sum_{i=0}^{n} T_i, \\ &= T_{n+1} + (T_n + T_{n+2} - 1)/2, \\ &= (2T_{n+1} + T_n + T_{n+2} - 1)/2, \\ &= (T_{n+1} + (T_{n+2} + T_{n+1} + T_n) - 1)/2, \\ &= (T_{n+1} + T_{n+3} - 1)/2 \end{aligned}$$

Lemma B.1.7 $T_n = (1/2^k) \sum_{j=0}^{k} \binom{k}{j} T_{n+k-4j}$, *for all* $k \geq 1$.

Proof. We shall prove this lemma by induction. The base case ($k = 1$),

$$T_n = (T_{n-3} + T_{n+1})/2,$$

is immediately obvious. Now suppose

$$T_n = (1/2^k) \sum_{j=0}^{k} \binom{k}{j} T_{n+k-4j}.$$

Observe that the base case gives us the following,

$$T_{n+k-4j} = (T_{n+k-4j-3} + T_{n+k-4j+1})/2.$$

Substituting the above expression we get

$$T_n = (1/2^{k+1}) \sum_{j=0}^{k} \binom{k}{j} (T_{n+k-4j-3} + T_{n+k-4j+1})$$

$$2^{k+1} T_n = \sum_{j=0}^{k} \binom{k}{j} T_{n+k-4j-3} + \sum_{j=0}^{k} \binom{k}{j} T_{n+k-4j+1}$$

Now we replace

$$\sum_{j=0}^{k} \binom{k}{j} T_{n+k-4j-3}$$

with

$$\sum_{j=1}^{k+1} \binom{k}{j-1} T_{n+k-4j+1}$$

to get

$$2^{k+1} T_n = \sum_{j=1}^{k+1} \binom{k}{j-1} T_{n+k-4j+1} + \sum_{j=0}^{k} \binom{k}{j} T_{n+k-4j+1}$$

Rewriting the summations so that we have the same bounds for each we get,

$$\begin{aligned} 2^{k+1} T_n &= \binom{k}{0} T_{n+k+1} + \sum_{j=1}^{k} \binom{k}{j-1} T_{n+k-4j+1} \\ &+ \sum_{j=1}^{k} \binom{k}{j} T_{n+k-4j+1} + \binom{k}{k} T_{n+k-4(k+1)+1}. \\ 2^{k+1} T_n &= \binom{k}{0} T_{n+k+1} \\ &+ \sum_{j=1}^{k} (\binom{k}{j-1} + \binom{k}{j}) T_{n+k-4j+1} \\ &+ \binom{k}{k} T_{n+k-4(k+1)+1}. \end{aligned}$$

Now we get a bit clever and notice that we can make the following substitutions in the equation above:

$$\begin{aligned}\binom{k}{0} &= \binom{k+1}{0},\\ \binom{k}{k} &= \binom{k+1}{k+1},\\ \binom{k}{j-1}+\binom{k}{j} &= \binom{k+1}{j}\end{aligned}$$

and arrive at

$$2^{k+1}T_n = \binom{k+1}{0}T_{n+k+1} + (\sum_{j=1}^{k}\binom{k+1}{j}T_{n+k-4j+1}) + \binom{k+1}{k+1}T_{n+k-4(k+1)+1}.$$

which is the same as

$$2^{k+1}T_n = \sum_{j=0}^{k+1}\binom{k+1}{j}T_{n+k-4j+1}.$$

Indeed, this proves the lemma.

The following lemmas can be easily proven using the above results, specifically lemma B.1.3 and lemma B.1.5; the proofs are little more than exercises in algebraic manipulation.

Lemma B.1.8 *If $T_n = a^{\alpha}b_1$ and $T_{n-1} = a^{\alpha-1}b_2$, then $T_{nk} = a^{\alpha}b_3$ and $T_{nk-1} = a^{\alpha-1}b_4, \forall k \geq 1$.*

Lemma B.1.9 *Suppose $n = 2^k w$, where w is odd and $k \geq 3$. Then $T_n = 2^{k-1}C$, where C is odd.*

While lemma B.1.8 and lemma B.1.9 could potentially aid in the factorization of Tribonacci numbers, it is an unfortunate truth that the only factors that follow this pattern are small.

An interesting property of the Tribonacci numbers we discovered (although again, nearly useless in terms of factorization) is,

$$2^{n-1} \mid T_{2^n},$$

and,

$$2^{n} \mid T_{2^n-1},$$

for some constants $C1$ and $C2$. This results follow directly from lemmas B.1.8 and B.1.9.

References

[1] E. Bach and J. Shallit. *Algorithmic Number Theory*, volume 1. The MIT Press, 1996.

[2] R. Brent. An improved monte carlo factorization algorithm. *Nordisk Tidskrift for Informationsbehandling (BIT)*, 20:176–184, 1980.

[3] R. Brent. Factorization of the tenth and elventh fermat numbers. Submitted to *Mathematics of Computation*, 1996.

[4] G. Wambach F. Damm, F. Heider. Mimd-factorisation on hypercubes. Technical Report 149, *Angewandte Mathematik Und Informatik Universität Zu Köln, 1994.*

[5] H. Wettig G. Wambach. Block sieving alogrithms. Technical Report 190, Angewandte Mathematik Und Informatik Universität Zu Köln, 1995.

[6] E. M. Wright G.H. Hardy. An Introduction to the Theory of Numbers. *Oxford Science Publications, 1995.*

[7] I. N. Herstein. Abstract Algebra. *Macmillan, 1986.*

[8] P. L. Montgomery J. Brillhart and R. D. Silverman. Tables of fibonacci and lucas factorizations. Mathematics of Computation, *50:251–260, 1988.*

[9] D. E. Knuth. The Art of Computer Programming, *volume 2. Addison-Wesley, 1981.*

[10] LiDIA-Group. LiDIA Manual a library for computational number theory, *3 1997. version 1.3.*

[11] R. Peralta. A quadratic sieve on the n-dimensional cube. Advances in Cryptology, Crypto '92, Lecture Notes in Computer Science 740, *pages 324–332, 1993.*

[12] J. M. Pollard. A monte carlo method of factorization. Nordisk Tidskrift for Informationsbehandling (BIT), *15:331–334, 1975.*

[13] Carl Pomerance. The quadratic sieve factoring algorithm. Advances of Cryptology, Proceedings of EuroCrypt 84, *pages 169–182, 1985.*

[14] Hans Riesel. Prime Numbers and Computer Methods for Factorization. *Birkhauser, 1994.*

[15] R. Silverman. The multiple polynomial quadratics sieve. Mathematics of Computation, *48:329–359, 1987.*

[16] S. Vajda. Fibonacci and Lucas Numbers, and the Golden Section. *Halsted Press, 1989.*

[17] N. Vorobev. Fibonacci Numbers. *Heath and Company, 1963.*

Supplement - Factoring Second and Third Order Recurrence Relations

Kirk Ocke

17 September 1997

1 Fibonacci Number Factorizations

$U_3 = [2]$

$U_4 = [3]$

$U_5 = [5]$

$U_6 = [2^3]$

$U_7 = [13]$

$U_8 = [3, 7]$

$U_9 = [2, 17]$

$U_{10} = [5, 11]$

$U_{11} = [89]$

$U_{12} = [2^4, 3^2]$

$U_{13} = [233]$

$U_{14} = [13, 29]$

$U_{15} = [2, 5, 61]$

$U_{16} = [3, 7, 47]$

$U_{17} = [1597]$

$U_{18} = [2^3, 17, 19]$

$U_{19} = [37, 113]$

$U_{20} = [3, 5, 11, 41]$

$U_{21} = [2, 13, 421]$

$U_{22} = [89, 199]$

$U_{23} = [28657]$

$U_{24} = [2^5, 3^2, 7, 23]$

$U_{25} = [5^2, 3001]$

$U_{26} = [233, 521]$

$U_{27} = [2, 17, 53, 109]$

$U_{28} = [3, 13, 29, 281]$

$U_{29} = [514229]$

$U_{30} = [2^3, 5, 11, 31, 61]$

$U_{31} = [557, 2417]$

$U_{32} = [3, 7, 47, 2207]$

$U_{33} = [2, 89, 19801]$

$U_{34} = [1597, 3571]$

$U_{35} = [5, 13, 141961]$

$U_{36} = [2^4, 3^3, 17, 19, 107]$

$U_{37} = [73, 149, 2221]$

$U_{38} = [37, 113, 9349]$

$U_{39} = [2, 233, 135721]$

$U_{40} = [3, 5, 7, 11, 41, 2161]$

$U_{41} = [2789, 59369]$

$U_{42} = [2^3, 13, 29. 211, 421]$

$U_{43} = [433494437]$

$U_{44} = [3, 43, 89, 199, 307]$

$U_{45} = [2, 5, 17, 61. 109441]$

$U_{46} = [139, 461, 28657]$

$U_{47} = [2971215073]$

$U_{48} = [2^6, 3^2. 7, 23, 47. 1103]$

$U_{49} = [13. 97, 6168709]$

$U_{50} = [5^2, 11. 101. 151. 3001]$

$U_{51} = [2, 1597. 6376021]$

$U_{52} = [3. 233, 521. 90481]$

$U_{53} = [953, 55945741]$

$U_{54} = [2^3, 17. 19. 53. 109. 5779]$

$U_{55} = [5, 89, 661. 474541]$

$U_{56} = [3. 7^2, 13, 29. 281, 14503]$

$U_{57} = [2, 37, 113. 797, 54833]$

$U_{58} = [59, 19489, 514229]$

$U_{59} = [353, 2710260697]$

$U_{60} = [2^4, 3^2, 5, 11, 31, 41. 61. 2521]$

$U_{61} = [4513, 555003497]$

$U_{62} = [557. 2417, 3010349]$

$U_{63} = [2, 13, 17, 421, 35239681]$

$U_{64} = [3. 7, 47. 1087. 2207. 4481]$

$U_{65} = [5. 233. 1473620616 1]$

$U_{66} = [2^3. 89. 199. 9901, 19801]$

$U_{67} = [269. 116849. 1429913]$

$U_{68} = [3, 67, 1597. 3571. 63443]$

$U_{69} = [2. 137, 829. 18077. 28657]$

$U_{70} = [5. 11. 13, 29. 71. 911. 141961]$

$U_{71} = [6673, 46165371073]$

$U_{72} = [2^5. 3^3. 7, 17. 19. 23. 107. 103681]$

$U_{73} = [9375829, 86020717]$

$U_{74} = [73, 149, 2221, 54018521]$

$U_{75} = [2, 5^2, 61, 3001. 230686501]$

$U_{76} = [3, 37, 113. 9349, 29134601]$

$U_{77} = [13. 89, 988681, 4832521]$

$U_{78} = [2^3, 79, 233, 521, 859, 135721]$

$U_{79} = [157, 92180471494753]$

$U_{80} = [3, 5, 7, 11, 41, 47, 1601, 2161, 3041]$

$U_{81} = [2, 17, 53, 109, 2269, 4373, 19441]$

$U_{82} = [2789, 59369, 370248451]$

$U_{83} = [99194853094755497]$

$U_{84} = [2^4, 3^2, 13, 29, 83, 211, 281, 421, 1427]$

$U_{85} = [5, 1597, 9521, 3415914041]$

$U_{86} = [6709, 144481, 433494437]$

$U_{87} = [2, 173, 514229, 3821263937]$

$U_{88} = [3, 7, 43, 89, 199, 263, 307, 881, 967]$

$U_{89} = [1069, 1665088321800481]$

$U_{90} = [2^3, 5, 11, 17, 19, 31, 61, 181, 541, 109441]$

$U_{91} = [13^2, 233, 741469, 159607993]$

$U_{92} = [3, 139, 461, 4969, 28657, 275449]$

$U_{93} = [2, 557, 2417, 4531100550901]$

$U_{94} = [2971215073, 6643838879]$

$U_{95} = [5, 37, 113, 761, 29641, 67735001]$

$U_{96} = [2^7, 3^2, 7, 23, 47, 769, 1103, 2207, 3167]$

$U_{97} = [193, 389, 3084989, 361040209]$

$U_{98} = [13, 29, 97, 6168709, 599786069]$

$U_{99} = [2, 17, 89, 197, 19801, 18546805133]$

$U_{100} = [3, 5^2, 11, 41, 101, 151, 401, 3001, 570601]$

$U_{101} = [743519377, 770857978613]$

$U_{102} = [2^3, 919, 1597, 3469, 3571, 6376021]$

$U_{103} = [519121, 5644193, 512119709]$

$U_{104} = [3, 7, 103, 233, 521, 90481, 102193207]$

$U_{105} = [2, 5, 13, 61, 421, 141961, 8288823481]$

$U_{106} = [953, 55945741, 119218851371]$

$U_{107} = [1247833, 8242065050061761]$

$U_{108} = [2^4, 3^4, 17, 19, 53, 107, 109, 5779, 11128427]$

$U_{109} = [827728777, 32529675488417]$

$U_{110} = [5, 11^2, 89, 199, 331, 661, 39161, 474541]$

$U_{111} = [2, 73, 149, 2221, 1459000305513721]$

$U_{112} = [3, 7^2, 13, 29, 47, 281, 14503, 10745088481]$

$U_{113} = [677, 272602401466814027129]$

$U_{114} = [2^3, 37, 113, 229, 797, 9349, 54833, 95419]$

$U_{115} = [5, 1381, 28657, 2441738887963981]$

$U_{116} = [3, 59, 347, 19489, 514229, 1270083883]$

$U_{117} = [2, 17, 233, 29717, 135721, 39589685693]$

$U_{118} = [353, 709, 8969, 336419, 2710260697]$

$U_{119} = [13, 1597, 159512939815855788121]$

$U_{120} = [2^5, 3^2, 5, 7, 11, 23, 31, 41, 61, 241, 2161, 2521, 20641]$

$U_{121} = [89, 97415813466381445596089]$

$U_{122} = [4513, 555003497, 5600748293801]$

$U_{123} = [2, 2789, 59369, 68541957733949701]$

$U_{124} = [3, 557, 2417, 3010349, 3020733700601]$

$U_{125} = [5^3, 3001, 158414167964045700001]$

$U_{126} = [2^3, 13, 17, 19, 29, 211, 421, 1009, 31249, 35239681]$

$U_{127} = [27941, 5568053048227732210073]$

$U_{128} = [3, 7, 47, 127, 1087, 2207, 4481, 186812208641]$

$U_{129} = [2, 257, 5417, 8513, 39639893, 433494437]$

$U_{130} = [5, 11, 131, 233, 521, 2081, 24571, 14736206161]$

$U_{131} = [106634041749171059581457216 9]$

$U_{132} = [2^4, 3^2, 43, 89, 199, 307, 9901, 19801, 261399601]$

$U_{133} = [13, 37, 113, 3457, 42293, 351301301942501]$

$U_{134} = [269, 4021, 116849, 1429913, 24994118449]$

$U_{135} = [2, 5, 17, 53, 61, 109, 109441, 111476995 4367361]$

$U_{136} = [3, 7, 67, 1597, 3571, 63443, 23230657239121]$

$U_{137} = [191347024000932780814494239 17]$

$U_{138} = [2^3, 137, 139, 461, 691, 829, 18077, 28657, 1485571]$

$U_{139} = [277, 2114537501, 855267229376890 93]$

$U_{140} = [3, 5, 11, 13, 29, 41, 71, 281, 911, 141961, 12317523121]$

$U_{141} = [2, 108289, 1435097, 142017737, 2971215073]$

$U_{142} = [6673, 46165371073, 688846502588399]$

$U_{143} = [89, 233, 8581, 192958415375685049662 1]$

$U_{144} = [2^6, 3^3, 7, 17, 19, 23, 47, 107, 1103, 103681, 10749957121]$

$U_{145} = [5, 514229, 3496199969307370798902 01]$

$U_{146} = [151549, 9375829, 86020717, 11899937029]$

$U_{147} = [2, 13, 97, 293, 421, 3529, 6168709, 347502052673]$

$U_{148} = [3, 73, 149, 2221, 11987, 54018521, 81143477963]$

$U_{149} = [110557, 162709, 4000949, 85607646594577]$

$U_{150} = [2^3, 5^2, 11, 31, 61, 101, 151, 3001, 12301, 18451, 230686501]$

$U_{151} = [5737, 28116666245258116464699158 77]$

$U_{152} = [3, 7, 37, 113, 9349, 29134601, 1091346396980401]$

$U_{153} = [2, 17^2, 1597, 6376021, 7175323114950564593]$

$U_{154} = [13, 29, 89, 199, 229769, 988681, 4832521, 9321929]$

$U_{155} = [5, 557, 2417, 21701, 12370533881, 61182778621]$

$U_{156} = [2^4, 3^2, 79, 233, 521, 859, 90481, 135721, 12280217041]$

$U_{157} = [313, 11617, 7636481, 10424204306491346737]$

$U_{158} = [157, 92180471494753, 32361122672259149]$

$U_{159} = [2, 317, 953, 55945741, 97639037, 229602768949]$

$U_{160} = [3, 5, 7, 11, 41, 47, 1601, 2161, 2207, 3041, 23725145626561]$

$U_{161} = [13, 8693, 28657, 612606107755058997065597]$

$U_{162} = [2^3, 17, 19, 53, 109, 2269, 3079, 4373, 5779, 19441, 62650261]$

$U_{163} = [977, 4892609, 33365519393, 32566223208133]$

$U_{164} = [3, 163, 2789, 59369, 800483, 350207569, 370248451]$

$U_{165} = [2, 5, 61, 89, 661, 19801, 86461, 474541, 518101, 900241]$

$U_{166} = [35761381, 6202401259, 99194853094755497]$

$U_{167} = [18104700793, 1966344318693345608565721]$

$U_{168} = [2^5, 3^2, 7^2, 13, 23, 29, 83, 167, 211, 281, 421, 1427, 14503, 65740583]$

$U_{169} = [233, 337, 89909, 104600155609, 126213229732669]$

$U_{170} = [5, 11, 1597, 3571, 9521, 1158551, 12760031, 3415914041]$

$U_{171} = [2, 17, 37, 113, 797, 6841, 54833, 5741461760879844361]$

$U_{172} = [3, 6709, 144481, 433494437, 313195711516578281]$

$U_{173} = [1639343785721, 3896787490076292715327 33]$

$U_{174} = [2^3, 59, 173, 349, 19489, 514229, 947104099, 3821263937]$

$U_{175} = [5^2, 13, 701, 3001, 141961, 1723120373020118930 8301]$

$U_{176} = [3, 7, 43, 47, 89, 199, 263, 307, 881, 967, 93058241, 562418561]$

$U_{177} = [2, 353, 2191261, 805134061, 129702768 1, 2710260697]$

$U_{178} = [179, 1069, 166508832180048 1, 2223550264098 8369]$

$U_{179} = [21481, 156089, 34188166409038989295346 13769]$

$U_{180} = [2^4, 3^3, 5, 11, 17, 19, 31, 41, 61, 107, 181, 541, 2521, 109441, 10783342081]$

$U_{181} = [8689, 422453, 81757892372385475745514611093]$

$U_{182} = [13^2, 29, 233, 521, 741469, 159607993, 6896671519701 61]$

$U_{183} = [2, 1097, 4513, 555003497, 1429734797197575 78008833]$

$U_{184} = [3, 7, 139, 461, 4969, 28657, 253367, 275449, 9506372193863]$

$U_{185} = [5, 73, 149, 2221, 17029455131913055569070976181 61]$

$U_{186} = [2^3, 557, 2417, 63799, 3010349, 35510749, 4531100550901]$

$U_{187} = [89, 373, 1597, 10157807305963434099105034917037]$

$U_{188} = [3, 563, 5641, 2971215073, 6643838879, 4632894751907]$

$U_{189} = [2, 13, 17, 53, 109, 421, 38033, 35239681, 9559219503167 35037]$

$U_{190} = [5, 11, 37, 113, 191, 761, 9349, 29641, 41611, 67735001, 87382901]$

$U_{191} = [4870723671313, 7578108062569891284390975793]$

$U_{192} = [2^8, 3^2, 7, 23, 47, 769, 1087, 1103, 2207, 3167, 4481, 11862575248703]$

$U_{193} = [9465278929, 10209304320323269339768260084 97]$

$U_{194} = [193, 389, 3299, 3084989, 361040209, 5667855750214 1579]$

$U_{195} = [2, 5, 61, 233, 135721, 14736206161, 8899925083749987 7681]$

$U_{196} = [3, 13, 29, 97, 281, 5881, 6168709, 599786069, 61025309469041]$

$U_{197} = [15761, 25795969, 227150265697, 717185107125886549]$

$U_{198} = [2^3, 17, 19, 89, 197, 199, 991, 2179, 9901, 19801, 1513909, 18546805133]$

$U_{199} = [397, 436782169201002048261171378550055269633]$

$U_{200} = [3, 5^2, 7, 11, 41, 101, 151, 401, 2161, 3001, 570601, 9125201, 5738108801]$

$U_{201} = [2, 269, 116849, 1429913, 505026070439624716931599 9021]$

$U_{202} = [809, 7879, 743519377, 770857978613, 201062946718741]$

$U_{203} = [13, 1217, 514229, 56470541, 258698270065673399465953 3]$

$U_{204} = [2^4, 3^2, 67, 409, 919, 1597, 3469, 3571, 63443, 6376021, 66265118449]$

$U_{205} = [5, 821, 2789, 59369, 125598581, 36448117857891321536401]$

$U_{206} = [619, 1031, 519121, 5644193, 512119709, 5257480026438961]$

$U_{207} = [2, 17, 137, 829, 18077, 28657, 407235315577362760122219648 1]$

$U_{208} = [3, 7, 47, 103, 233, 521, 3329, 90481, 102193207, 106513889, 325759201]$

$U_{209} = [37, 89, 113, 57314120955051297736679165379998262001]$

$U_{210} = [2^3, 5, 11, 13, 29, 31, 61, 71, 211, 421, 911, 21211, 141961, 767131, 8288823481]$

$U_{211} = [22504837, 38490197, 800972881, 80475423858449593021]$

$U_{212} = [3, 953, 1483, 2969, 55945741, 119218851371, 1076012367720403]$

$U_{213} = [2, 1277, 6673, 46165371073, 185790722054921374395775013]$

$U_{214} = [1247833, 8242065050061761, 47927441, 479836483312919]$

$U_{215} = [5, 433494437, 2607553541, 67712817361580804952011621]$

$U_{216} = [2^5, 3^4, 7, 17, 19, 23, 53, 107, 109, 5779, 6263, 103681, 11128427, 177962167367]$

$U_{217} = [13, 433, 557, 2417, 44269, 217221773, 219117486 1, 6274653314021]$

$U_{218} = [128621, 788071, 827728777, 32529675488417, 593985111211]$

$U_{219} = [2, 123953, 9375829, 86020717, 4139537, 3169251245945843761]$

$U_{220} = [3, 5, 11^2, 41, 43, 89, 199, 307, 331, 661, 39161, 474541, 59996854928656801]$

$U_{221} = [233, 1597, 203572412497, 90657498718024645332639294 0193]$

$U_{222} = [2^3, 73, 149, 2221, 4441, 146521, 1121101, 54018521, 1459000305513721]$

$U_{223} = [4013, 108377, 251534189, 16434461004641013899615607 0813]$

$U_{224} = [3, 7^2, 13, 29, 47, 223, 281, 449, 2207, 14503, 10745088481, 1154149773784223]$

$U_{225} = [2, 5^2, 17, 61, 3001, 109441, 230686501, 11981661982050957053616001]$

$U_{226} = [677, 272602401466814027129, 41267042784492103747077 1]$

$U_{227} = [23609, 52195341379830251590788471136194672857273 77]$

$U_{228} = [2^4, 3^2, 37, 113, 227, 229, 797, 9349, 26449, 54833, 95419, 29134601, 212067587]$

$U_{229} = [457, 2749, 40487201, 13260544901, 17831560297620361798553]$

$U_{230} = [5, 11, 139, 461, 1151, 1381, 5981, 28657, 324301, 686551, 2441738887963981]$

$U_{231} = [2, 13, 89, 421, 19801, 988681, 1832521, 91642596017 8159235188401]$

$U_{232} = [3, 7, 59, 347, 19489, 299281, 514229, 1270083883, 83412841087950 6721]$

$U_{233} = [139801, 25047390419633, 63148408958396311955782954 7141]$

$U_{234} = [2^3, 17, 19, 79, 233, 521, 859, 29717, 135721, 3958906856 93, 1052645985555841]$

$U_{235} = [5, 2971215073, 389678426275593986752662955603693114561]$

$U_{236} = [3, 353, 709, 8969, 336419, 15247723, 2710260697, 100049587197598387]$

$U_{237} = [2, 157, 1668481, 40762577, 92180471494753, 7698999052751136773]$

$U_{238} = [13, 29, 239, 1597, 3571, 10711, 27932732439809, 159512939815855788121]$

$U_{239} = [10037, 62141, 2228536579597318057, 28546908862296149233369]$

$U_{240} = [2^6, 3^2, 5, 7, 11, 23, 31, 41, 47, 61, 241, 1103, 1601, 2161, 2521, 3041, 20641, 23735900452321]$

$U_{241} = [11042621, 7005329677, 13428748928964476326795282473927 3]$

$U_{242} = [89, 199, 9741581346638144559608 9, 9742073320849186904419 9]$

$U_{243} = [2, 17, 53, 109, 2269, 4373, 19441, 448607550257, 16000411124306403070561]$

$U_{244} = [3, 4513, 19763, 21291929, 555003497, 5600748293801, 24848660119363]$

$U_{245} = [5, 13, 97, 141961, 6168709, 128955073914024460192651484843195641]$

$U_{246} = [2^3, 2789, 59369, 4767481, 370248451, 7188487771, 68541957733949701]$

$U_{247} = [37, 113, 233, 409100738617, 467730604336790467692631214732 8153]$

$U_{248} = [3, 7, 557, 743, 2417, 467729, 3010349, 3020733700601, 33758740830460183]$

$U_{249} = [2, 1033043205255409, 99194853004755497, 23812215284009787769]$

$U_{250} = [5^3, 11, 101, 151, 251, 3001, 112128001, 28143378001, 158414167964045700001]$

$U_{251} = [582416774750273, 219370803294651220261871241996569619 13]$

$U_{252} = [2^4, 3^3, 13, 17, 19, 29, 83, 107, 211, 281, 421, 1009, 1427, 31249, 1461601, 35239681, 764940961]$

$U_{253} = [89, 28657, 4322114369, 2201228230611589, 13784979303338047612061]$

$U_{254} = [509, 5081, 27941, 487681, 1382261, 19954241, 5568053048227732210073]$

$U_{255} = [2, 5, 61, 1597, 9521, 6376021, 311591 4041, 207786443969 41, 20862774425341]$

$U_{256} = [3, 7, 47, 127, 1087, 2207, 4481, 112889, 186812208641, 469816763452337 9875583]$

$U_{257} = [5653, 3297197867164590565 45521, 123002672171931341713607 14649]$

$U_{258} = [2^3, 257, 5417, 6709, 8513, 144481, 308311, 39639893, 433494437, 761882591401]$

$U_{259} = [13, 73, 149, 1553, 2221, 4046567733793, 30112667422957719851 48799223649]$

$U_{260} = [3, 5, 11, 41, 131, 233, 521, 2081, 3121, 21571, 90481, 14736206161, 42426476041450801]$

$U_{261} = [2, 17, 173, 2089, 20357, 36017, 40993, 322073, 514229, 3821263937, 6857029027549]$

$U_{262} = [1049, 4149886984 61, 5177332620991, 10663401174917105958 14572169]$

$U_{263} = [4733, 93629, 928362296 0 63901912352912 0 6 8 1425664630809390281]$

$U_{264} = [2^5, 3^2, 7, 23, 43, 89, 199, 263, 307, 881, 967, 5281, 9901, 19801, 66529, 152204449, 261399601]$

$U_{265} = [5, 953, 15901, 5591571 1, 2741218753 3681, 92691859915746812592 0827581]$

$U_{266} = [13, 29, 37, 113, 3137, 9349, 12293, 100941121739, 215295865045 9, 351301301942501]$

$U_{267} = [2, 1069, 122887 12515329 9, 166508832180018 1, 6145587734970304287 7309]$

$U_{268} = [3, 269, 4021, 6163, 116849, 1129913, 2499111 8449, 201912469249, 2705622682163]$

$U_{269} = [5381, 251797518266981 3, 321709147478106 41, 16936043982964 8789853]$

$U_{270} = [2^3, 5, 11, 17, 19, 31, 53, 61, 109, 181, 271, 541, 811, 5779, 42391, 109441, 119611, 1114769954367361]$

$U_{271} = [4491870763 48273, 4302672125258671219517406190935941938058573]$

$U_{272} = [3, 7, 47, 67, 1597, 3571, 63443, 232306572301 21, 5626278372832919401376548 81]$

$U_{273} = [2, 13^2, 233, 421, 135721, 640457, 741469, 159607993, 1483547330343905886515273]$

$U_{274} = [541721291, 7898248787093905828 1, 191347024000932780814494239 17]$

$U_{275} = [5^2, 89, 661, 3001, 474541, 7239101, 15806979101, 5527278404454199535821801]$

$U_{276} = [2^4, 3^2, 137, 139, 461, 691, 829, 4969, 16561, 18077, 28657, 162563, 275449, 1485571, 1043766587]$

$U_{277} = [505471005740691524853293621, 68611213081873309089863281049 17]$

$U_{278} = [277, 30859, 2114537501, 85526722937689093, 253279129, 14331800109223159]$

$U_{279} = [2, 17, 557, 2417, 11717, 4531100550901, 594960058508093, 6279830532252706321]$

$U_{280} = [3, 5, 7^2, 11, 13, 29, 41, 71, 281, 911, 2161, 14503, 141961, 12317523121, 118021448662479038881]$

$U_{281} = [174221, 119468273, 114205973520041784262049438829321530369345505 7]$

$U_{282} = [2^3, 108289, 1435097, 2971215073, 79099591, 142017737, 6643838879, 139509555271]$

$U_{283} = [10753, 825229, 15791401, 4441118888488058431632357842986308632648 81]$

$U_{284} = [3, 283, 569, 6673, 2820403, 9799987, 35537616083, 46165371073, 688846502588399]$

$U_{285} = [2, 5, 37, 61, 113, 761, 797, 29641, 54833, 67735001, 95673461671504632850248033060 1]$

$U_{286} = [89, 199, 233, 521, 8581, 1957099, 2120119, 1784714380021, 19295841537568504966 21]$

$U_{287} = [13, 2789, 59369, 19816007100185326779670069250749018457050106438220 1]$

$U_{288} = [2^7, 3^3, 7, 17, 19, 23, 47, 107, 769, 1103, 2207, 3167, 103681, 10749957121, 115561578124838522881]$

$U_{289} = [577, 1597, 1733, 98837, 101232653, 106205194357, 65807865827772544448384854 1]$

$U_{290} = [5, 11, 59, 19489, 514229, 120196353941, 1322154751061, 349619996930737079890201]$

$U_{291} = [2, 193, 389, 3084989, 361040209, 76674415738994499773, 2279931177549758706 77]$

$U_{292} = [3, 29201, 151549, 9375829, 86020717, 11899937029, 37125857850184727260788881]$

$U_{293} = [64390759997, 11886939163497285252295209896447615523813499731472 9]$

$U_{294} = [2^3, 13, 29, 97, 211, 293, 421, 3529, 65269, 620929, 6168709, 8844991, 599786069, 347502052673]$

$U_{295} = [5, 353, 1181, 35401, 75521, 160481, 737501, 2710260697, 1120969250625390660846912 1]$

$U_{296} = [3, 7, 73, 149, 2221, 11987, 10661921, 54018521, 81143477963, 1140872880487019539984 01]$

$U_{297} = [2, 17, 53, 89, 109, 197, 593, 4157, 19801, 1360418597, 18546805133, 12369243068750242280033]$

$U_{298} = [110557, 162709, 952111, 4000949, 856076465945 77, 4434539, 3263039535803245519]$

$U_{299} = [233, 28657, 2056992877234275208463485342027139282056040284860517152 1]$

$U_{300} = [2^4, 3^2, 5^2, 11, 31, 41, 61, 101, 151, 401, 601, 2521, 3001, 12301, 18451, 570601, 230686501, 87129547172401]$

$U_{301} = [13, 433494437, 6380692745271404734077815684636927896943536596672852 1]$

$U_{302} = [1511, 5737, 109734721, 2175330001848357747 79, 2811666624525811646469915877]$

$U_{303} = [2, 743519377, 770857978613, 85502243896744 81, 9604965791727987485136942 1]$

$U_{304} = [3, 7, 37, 47, 113, 9349, 29134601, 562766385967, 1091346396980401, 2206456200865197103]$

$U_{305} = [5, 2441, 4513, 6101, 555003497, 20415253966247698801, 6472776707179982409438 61]$

$U_{306} = [2^3, 17^2, 19, 919, 1597, 3469, 3571, 13159, 6376021, 8293976826829399, 7175323114950564593]$

$U_{307} = [613, 9143689, 5307027867738937, 2169138415139880143903925835206814718 57]$

$U_{308} = [3, 13, 29, 43, 89, 199, 281, 307, 229769, 988681, 4832521, 9321929, 15252467, 900164950225760603]$

$U_{309} = [2, 617, 318889, 519121, 5644193, 512119709, 32386142297, 8833645636274593230408 61]$

$U_{310} = [5, 11, 311, 557, 2417, 21701, 3010349, 12370533881, 29138888651, 61182778621, 823837075741]$

$U_{311} = [837833, 6872477, 603717553, 1272232704013218608925801029523104780183 8093]$

$U_{312} = [2^5, 3^2, 7, 23, 79, 103, 233, 521, 859, 1249, 90481, 135721, 102193207, 12280217041, 94491842183551489]$

$U_{313} = [1877, 5009, 7901346123803597, 155858325142313284065543879972611970587627 3]$

$U_{314} = [313, 11617, 7636481, 10424204306491346737, 39980051, 161888565752865178188491 71]$

$U_{315} = [2, 5, 13, 17, 61, 421, 109441, 141961, 9761221, 35239681, 8288823481, 120570028745492370271501]$

$U_{316} = [3, 157, 21803, 5924683, 14629892449, 184715524801, 92180471494753, 32361122672259149]$

$U_{317} = [1307309, 503546330165333805042385219 09, 120553346549469824534649942768 37]$

$U_{318} = [2^3, 317, 953, 785461, 55945741, 97639037, 119218851371, 229602768949, 4523819299182451]$

$U_{319} = [89, 1913, 514229, 578029, 1435522969, 1535414556003613, 1862624318468346334828 3529]$

$U_{320} = [3, 5, 7, 11, 41, 47, 641, 1087, 1601, 2161, 2207, 3041, 4481, 23725145626561, 8781322404439748742016 01]$

$U_{321} = [2, 1247833, 8242065050061761, 2644387026552261937524585811210551514149289 21]$

$U_{322} = [13, 29, 139, 461, 1289, 8693, 28657, 1917511, 965840862268529759, 6126061077550589970655 97]$

$U_{323} = [37, 113, 1597, 1109581873, 85542646443577, 2246961751521627497245934985432764 2081]$

$U_{324} = [2^4, 3^5, 17, 19, 53, 107, 109, 2269, 3079, 4373, 5779, 19441, 11128427, 62650261, 1828620361, 6782976947987]$

$U_{325} = [5^2, 233, 1301, 3001, 4235401, 605416501, 880262501, 14736206161, 4928470696778756905830 1]$

$U_{326} = [977, 1043201, 4892609, 33365519393, 6601501, 32566223208133, 1686454671192230445929]$

$U_{327} = [2, 653, 827728777, 32529675488417, 1746181, 158954614142727267943384636436638045 7]$

$U_{328} = [3, 7, 163, 2789, 59369, 800483, 350207569, 370248451, 2684571411430027028247905903965201]$

$U_{329} = [13, 1973, 26321, 2971215073, 1273918744110975926724698913756444771419485730202 37]$

$U_{330} = [2^3, 5, 11^2, 31, 61, 89, 199, 331, 661, 9901, 19801, 39161, 86461, 474541, 518101, 900241, 51164521, 1550853481]$

$U_{331} = [29129, 2296668664863212027639122802848520084131849762253337059166450246 1]$

$U_{332} = [3, 35761381, 6202401259, 6464041, 245329617161, 10341247759646081, 99194853094755497]$

$U_{333} = [2, 17, 73, 149, 2221, 12653, 124134848933957, 1459000305513721, 9305077315575902267675937 61]$

$U_{334} = [766531, 18104700793, 1966344318693345608565721, 1038499276935845423201273279 09]$

$U_{335} = [5, 269, 116849, 1429913, 2040410654589510290615412852018689113300321765114476636 1]$

$U_{336} = [2^6, 3^2, 7^2, 13, 23, 29, 47, 83, 167, 211, 281, 421, 1103, 1427, 14503, 65740583, 10745088481, 115613939510481515041]$

$U_{337} = [673, 15229266824729, 1171266446222833267851409604104331211834067048447153001]$

$U_{338} = [233, 337, 521, 89909, 104600155609, 126213229732669, 596107814364089, 671040394220849329]$

$U_{339} = [2, 677, 149161, 258317, 2726024014668 14027129, 2209878650579776888742215348691420033]$

$U_{340} = [3, 5, 11, 41, 67, 1361, 1597, 3571, 9521, 40801, 63443, 1158551, 12760031, 3415914041, 11614654211954032961]$

$U_{341} = [89, 557, 2417, 761227665342913, 197907695243868721, 4558282384863830955384586674337]$

$U_{342} = [2^3, 17, 19^2, 37, 113, 229, 797, 6841, 9349, 54833, 95419, 162451, 1617661, 7038398989, 5741461760879844361]$

$U_{343} = [13, 97, 46649, 6168709, 5449038756620509, 1089441709440098759783067514822344147023 93]$

$U_{344} = [3, 7, 6709, 144481, 433494437, 313195711516578281, 126117711915911646784404045944033521]$

$U_{345} = [2, 5, 61, 137, 829, 1381, 18077, 28657, 186301, 2441738887963981, 2501386404496144797315281460498 1]$

$U_{346} = [78889, 1639343785721, 3896787490076292715327 33, 6248069, 16923049609, 171246170261359]$

$U_{347} = [324097, 1434497, 3345860598013, 34201673799023762317333, 2775212903578562218403359 3]$

$U_{348} = [2^4, 3^2, 59, 173, 347, 349, 19489, 97787, 514229, 947104099, 3821263937, 528295667, 1270083883, 5639710969]$

$U_{349} = [1358309, 2663569, 275209307376778770586 73, 3882181160843921539281398566047324725 3]$

$U_{350} = [5^2, 11, 13, 29, 71, 101, 151, 701, 911, 3001, 54601, 141961, 560701, 7517651, 51636551, 17231203730201189308301]$

$U_{351} = [2, 17, 53, 109, 233, 29717, 135721, 2623373, 8023861, 39589685693, 6579032167974049037174409803 4257]$

$U_{352} = [3, 7, 43, 47, 89, 199, 263, 307, 881, 967, 1409, 2207, 93058241, 562418561, 6086461133983, 319702847642258783]$

$U_{353} = [736357, 359802011012573915498603609235632625259749492479918321872573852016 89]$

$U_{354} = [2^3, 353, 709, 8969, 336419, 2191261, 805134061, 1297027681, 2710260697, 10884439, 10511761735170 6859]$

$U_{355} = [5, 4261, 6673, 75309701, 309273161, 46165371073, 9207609261398081, 492797226433918641928 01]$

$U_{356} = [3, 179, 1069, 1665088321800481, 22235502640988369, 528054453566747229127714911929654620 1]$

$U_{357} = [2, 13, 421, 1429, 1597, 258469, 6376021, 159512939815855788121, 276538662398362584638816230929 61]$

$U_{358} = [359, 21481, 156089, 1066737847220321, 66932254279484647441, 341881664090389892953461376 9]$

$U_{359} = [47542043773469822074736802716674938292770141701655719366226871637693547624 1]$

$U_{360} = [2^5, 3^3, 5, 7, 11, 17, 19, 23, 31, 41, 61, 107, 181, 241, 541, 2161, 2521, 8641, 20641, 103681, 109441, 10783342081, 13373763765986881]$

$U_{361} = [37, 113, 6567762529, 1196762644057, 3150927827816930878141597, 120201265107147347830092 41]$

$U_{362} = [8689, 97379. 422453, 21373261504197751, 32242356485644069, 817578923723854757455146109 3]$

$U_{363} = [2. 89, 19801. 97415813466381445596089, 94905596043359637960818476990353850018366158 01]$

$U_{364} = [3. 13^2, 29. 233, 281, 521, 90481, 232961, 741469, 159607993, 689667151970161, 611057863429488653480848 1]$

$U_{365} = [5. 210241. 9375829, 86020717, 27583781, 758275080626801, 481086261779233475625991833542941]$

$U_{366} = [2^3, 1097. 4513, 555003497, 14297347971975757800833, 14686239709, 5600748293801, 533975715909289]$

$U_{367} = [733. 17969789. 75991753, 5648966761, 43397676601, 114150315493, 797357235624701499134444201]$

$U_{368} = [3. 7, 47, 139, 367, 461. 4969, 28657, 253367, 275449, 9506372193863, 37309023160481, 4417209581003819171 03]$

$U_{369} = [2. 17. 2789. 8117, 59369, 199261. 84738793193, 9382599520669, 68541957733949701, 11783851863335146 9]$

$U_{370} = [5. 11, 73, 149. 2221. 54018521, 265272771839851, 2918000731816531, 17029455131913055569070976181 61]$

$U_{371} = [13. 953, 207017, 55945741, 10668914543306923609111184699154927702112864025685324579661 13]$

$U_{372} = [2^4, 3^2, 557. 2417. 63799, 3010349, 35510749. 15917507, 3020733700601, 4531100550901, 859886421593527043]$

$U_{373} = [2237. 9697, 371509, 20580649, 24164233642269551523833039687561541379284635421201183694 57]$

$U_{374} = [89. 199, 373. 1597. 1871. 3571, 90567423440850652626509739043 1, 10157807305963434099105034917037]$

$U_{375} = [2, 5^3, 61, 3001, 9001, 169501, 230686501, 41510105455501, 99062934069446535 01, 158414167964045700001]$

$U_{376} = [3. 7, 563, 5641. 18049, 100769, 2971215073, 6643838879, 4632894751907, 153037630649666194962091 443041]$

$U_{377} = [233, 514229. 104264251753, 361575655741, 60814658534556798167089319998544920201506009 4237]$

$U_{378} = [2^3, 13. 17. 19. 29, 53, 109, 211, 379, 421, 1009, 5779. 31249. 38933. 85429, 912871, 3523968 1, 955921950316735037, 1258740001]$

$U_{379} = [757, 11889989, 642738983137, 12431361016316631290799797824272547354748169053405045242 21]$

$U_{380} = [3, 5, 11, 37. 41. 113, 191, 761, 2281, 4561, 9349, 29641. 41611, 67735001, 87382901, 29134601, 782747561, 174795553490801]$

$U_{381} = [2. 27941. 18995897. 5568053048227732210073, 3185450213669826966828420712039093359617657693]$

$U_{382} = [22921. 4870723671313. 75781080625698912843997 5793, 395586472506832921, 910257559954057439]$

2 Lucas Number Factorizations

$V_3 = [2^2]$

$V_4 = [7]$

$V_5 = [11]$

$V_6 = [2, 3^2]$

$V_7 = [29]$

$V_8 = [47]$

$V_9 = [2^2, 19]$

$V_{10} = [3, 41]$

$V_{11} = [199]$

$V_{12} = [2, 7, 23]$

$V_{13} = [521]$

$V_{14} = [3, 281]$

$V_{15} = [2^2, 11, 31]$

$V_{16} = [2207]$

$V_{17} = [3571]$

$V_{18} = [2, 3^3, 107]$

$V_{19} = [9349]$

$V_{20} = [7, 2161]$

$V_{21} = [2^2, 29, 211]$

$V_{22} = [3, 43, 307]$

$V_{23} = [139, 461]$

$V_{24} = [2, 47, 1103]$

$V_{25} = [11, 101, 151]$

$V_{26} = [3, 90481]$

$V_{27} = [2^2, 19, 5779]$

$V_{28} = [7^2, 14503]$

$V_{29} = [59, 19489]$

$V_{30} = [2, 3^2, 41, 2521]$

$V_{31} = [3010349]$

$V_{32} = [1087, 4481]$

$V_{33} = [2^2, 199, 9901]$

$V_{34} = [3, 67, 63443]$

$V_{35} = [11, 29, 71, 911]$

$V_{36} = [2, 7, 23, 103681]$

$V_{37} = [54018521]$

$V_{38} = [3, 29134601]$

$V_{39} = [2^2, 79, 521, 859]$

$V_{40} = [47, 1601, 3041]$

$V_{41} = [370248451]$

$V_{42} = [2, 3^2, 83, 281, 1427]$

$V_{43} = [6709, 144481]$

$V_{44} = [7, 263, 881, 967]$

$V_{45} = [2^2, 11, 19, 31, 181, 541]$

$V_{46} = [3, 4969, 275449]$

$V_{47} = [6643838879]$

$V_{48} = [2, 769, 2207, 3167]$

$V_{49} = [29, 599786069]$

$V_{50} = [3, 41, 401, 570601]$

$V_{51} = [2^2, 919, 3469, 3571]$

$V_{52} = [7, 103, 102193207]$

$V_{53} = [119218851371]$

$V_{54} = [2, 3^4, 107, 11128427]$

$V_{55} = [11^2, 199, 331, 39161]$

$V_{56} = [47, 107458848 1]$

$V_{57} = [2^2, 229, 9349, 95419]$

$V_{58} = [3, 347, 1270083883]$

$V_{59} = [709, 8969, 336419]$

$V_{60} = [2, 7, 23, 241, 2161, 20641]$

$V_{61} = [5600748293801]$

$V_{62} = [3, 3020733700601]$

$V_{63} = [2^2, 19, 29, 211, 1009, 31249]$

$V_{64} = [127, 186812208641]$

$V_{65} = [11, 131, 521, 2081, 24571]$

$V_{66} = [2, 3^2, 43, 307, 261399601]$

$V_{67} = [4021, 249011 8449]$

$V_{68} = [7, 23230657239121]$

$V_{69} = [2^2, 139, 461, 691, 1485571]$

$V_{70} = [3, 41, 281, 12317523121]$

$V_{71} = [6888465025 88399]$

$V_{72} = [2, 47, 1103, 1074995721]$

$V_{73} = [151549, 1189993702 9]$

$V_{74} = [3, 11987, 81143477963]$

$V_{75} = [2^2, 11, 31, 101, 151, 12301, 18451]$

$V_{76} = [7, 109134639680401]$

$V_{77} = [29, 199, 229769, 9321929]$

$V_{78} = [2, 3^2, 90481, 12280217041]$

$V_{79} = [32361122672259149]$

$V_{80} = [2207, 23725145626561]$

$V_{81} = [2^2, 19, 3079, 5779, 62650261]$

$V_{82} = [3, 163, 800483, 350207569]$

$V_{83} = [35761381, 6202401259]$

$V_{84} = [2, 7^2, 23, 167, 14503, 65740583]$

$V_{85} = [11, 3571, 1158551, 12760031]$

$V_{86} = [3, 313195711516578281]$

$V_{87} = [2^2, 59, 349, 19489, 947104099]$

$V_{88} = [47, 93058241, 562418561]$

$V_{89} = [179, 22235502640988369]$

$V_{90} = [2, 3^3, 41, 107, 2521, 10783342081]$

$V_{91} = [29, 521, 689667151970161]$

$V_{92} = [7, 253367, 9506372193863]$

$V_{93} = [2^2, 63799, 3010349, 35510749]$

$V_{94} = [3, 563, 5641, 4632894751907]$

$V_{95} = [11, 191, 9349, 41611, 87382901]$

$V_{96} = [2, 1087, 4481, 11862575248703]$

$V_{97} = [3299, 5667855750214 1579]$

$V_{98} = [3, 281, 5881, 6102530946904 1]$

$V_{99} = [2^2, 19, 199, 991, 2179, 9901, 1513909]$

$V_{100} = [7, 2161, 9125201, 5738108801]$

$V_{101} = [809, 7879, 201062946718741]$

$V_{102} = [2, 3^2, 67, 409, 63443, 66265118449]$

$V_{103} = [619, 1031, 525748002643896 1]$

$V_{104} = [47, 3329, 106513889, 325759201]$

$V_{105} = [2^2, 11, 29, 31, 71, 211, 911, 21211, 767131]$

$V_{106} = [3, 1483, 2969, 1076012367720403]$

$V_{107} = [47927441, 4798364833129 19]$

$V_{108} = [2, 7, 23, 6263, 103681, 177962167367]$

$V_{109} = [128621, 788071, 59398511121 1]$

$V_{110} = [3, 41, 43, 307, 599968519286568 01]$

$V_{111} = [2^2, 4441, 146521, 1121101, 51018521]$

$V_{112} = [223, 449, 2207, 1154149573781223]$

$V_{113} = [4126704278449210374707 71]$

$V_{114} = [2, 3^2, 227, 26449, 29134601, 212067587]$

$V_{115} = [11, 139, 461, 1151, 5981, 324301, 686551]$

$V_{116} = [7, 299281, 8344284108795067 21]$

$V_{117} = [2^2, 19, 79, 521, 859, 10526459855555841]$

$V_{118} = [3, 15247723, 100049587197598387]$

$V_{119} = [29, 239, 3571, 10711, 27932732439809]$

$V_{120} = [2, 47, 1103, 1601, 3041, 23735900452321]$

$V_{121} = [199, 97420733208491869044199]$

$V_{122} = [3, 19763, 21291929, 24848660119363]$

$V_{123} = [2^2, 4767481, 370248451, 7188487771]$

$V_{124} = [7, 743, 467729, 33758740830460183]$

$V_{125} = [11, 101, 151, 251, 112128001, 28143378001]$

$V_{126} = [2, 3^3, 83, 107, 281, 1427, 1461601, 764940961]$

$V_{127} = [509, 5081, 487681, 13822681, 19954241]$

$V_{128} = [119809, 469816763452337987 5583]$

$V_{129} = [2^2, 6709, 144481, 308311, 761882591401]$

$V_{130} = [3, 41, 3121, 90481, 424264760411450801]$

$V_{131} = [1049, 414988698461, 5477332620091]$

$V_{132} = [2, 7, 23, 263, 881, 967, 5281, 66529, 152204449]$

$V_{133} = [29, 9349, 10694421739, 215205865015 9]$

$V_{134} = [3, 6163, 201912469249, 270562268216 3]$

$V_{135} = [2^2, 11, 19, 31, 181, 271, 541, 811, 5779, 42391, 119611]$

$V_{136} = [47, 562627837283291941D137654881]$

$V_{137} = [541721291, 7898248787093905828 1]$

$V_{138} = [2, 3^2, 4969, 16561, 162563, 275449, 1013766587]$

$V_{139} = [30859, 253279129, 1433180010922315 9]$

$V_{140} = [7^2, 2161, 14503, 118021448662479038881]$

$V_{141} = [2^2, 79099591, 6643838879, 13050955527 1]$

$V_{142} = [3, 283, 569, 2820403, 9799987, 3553761 6083]$

$V_{143} = [199, 521, 1957099, 2120119, 17817113 80021]$

$V_{144} = [2, 769, 2207, 3167, 11556137812183852 2881]$

$V_{145} = [11, 59, 19489, 1201963539011, 13221517 51061]$

$V_{146} = [3, 29201, 37125857850181727260788881]$

$V_{147} = [2^2, 29, 211, 65269, 620929, 881901, 5 99786069]$

$V_{148} = [7, 10661921, 1140872880187019539181 01]$

$V_{149} = [952111, 4134539, 3263039535803215519]$

$V_{150} = [2, 3^2, 41, 401, 601, 2521, 570601, 871295417172101]$

$V_{151} = [1511, 109734721, 21755330001818357754779]$

$V_{152} = [47, 562766385967, 22064562008651 97103]$

$V_{153} = [2^2, 19, 919, 3469, 3571, 13159, 82939768268293 9]$

$V_{154} = [3, 43, 281, 307, 15252467, 90D16195D225760603]$

$V_{155} = [11, 311, 30103419, 2913888851, 8238370755 41]$

$V_{156} = [2, 7, 23, 103, 1249, 102193207, 9449184218355 1489]$

$V_{157} = [39980051, 1618885657528651781884917 1]$

$V_{158} = [3, 21803, 5924683, 14629892449, 184715524801]$

$V_{159} = [2^2, 785461, 119218851371, 4523819299182451]$

$V_{160} = [641, 1087, 4481, 878132240443974874201601]$

$V_{161} = [29, 139, 461, 1289, 1917511, 965840862268529759]$

$V_{162} = [2, 3^5, 107, 11128427, 1828620361, 6782976947987]$

$V_{163} = [1043201, 6601501, 1686454671192230445929]$

$V_{164} = [7, 2684571411430027028247905903965201]$

$V_{165} = [2^2, 11^2, 31, 199, 331, 9901, 39161, 51164521, 1550853481]$

$V_{166} = [3, 6464041, 245329617161, 10341247759646081]$

$V_{167} = [766531, 10384992769358454232012732 7909]$

$V_{168} = [2, 47, 1103, 10745088481, 115613939510481515041]$

$V_{169} = [521, 596107814364089, 671040394220849329]$

$V_{170} = [3, 41, 67, 1361, 40801, 63443, 1161465421195403296 1]$

$V_{171} = [2^2, 19^2, 229, 9349, 95419, 162451, 1617661, 7038398989]$

$V_{172} = [7, 126117711915911646784404045944033521]$

$V_{173} = [78889, 6248069, 16923049609, 171246170261359]$

$V_{174} = [2, 3^2, 347, 97787, 528295667, 1270083883, 5639710969]$

$V_{175} = [11, 29, 71, 101, 151, 911, 54601, 560701, 7517651, 51636551]$

$V_{176} = [1409, 2207, 6086461133983, 319702847642258783]$

$V_{177} = [2^2, 709, 8969, 336419, 10884439, 105117617351706859]$

$V_{178} = [3, 52805445356674722912771491192965462 01]$

$V_{179} = [359, 1066737847220321, 66932254279484647441]$

$V_{180} = [2, 7, 23, 241, 2161, 8641, 20641, 103681, 13373763765986881]$

$V_{181} = [97379, 21373261504197751, 32242356485644069]$

$V_{182} = [3, 281, 90481, 232961, 611057863429488653480848 1]$

$V_{183} = [2^2, 14686239709, 5600748293801, 533975715909289]$

$V_{183} = [2^2, 14686239709, 5600748293801, 533975715909289]$

$V_{184} = [47, 367, 37309023160481, 4417209581003819171 03]$

$V_{185} = [11, 54018521, 265272771839851, 2918000731816531]$

3 G Number Factorizations

$G_4 = [2]$

$G_5 = [3]$

$G_6 = [2^2]$

$G_7 = [2, 3]$

$G_8 = [3^2]$

$G_9 = [13]$

$G_{10} = [19]$

$G_{11} = [2^2, 7]$

$G_{12} = [41]$

$G_{13} = [2^2, 3, 5]$

$G_{14} = [2^3, 11]$

$G_{15} = [3, 43]$

$G_{16} = [3^3, 7]$

$G_{17} = [277]$

$G_{18} = [2, 7, 29]$

$G_{19} = [5, 7, 17]$

$G_{20} = [2^3, 109]$

$G_{21} = [2, 3^2, 71]$

$G_{22} = [1873]$

$G_{23} = [3^2. 5, 61]$

$G_{24} = [3^3, 149]$

$G_{25} = [2^3, 11, 67]$

$G_{26} = [8641]$

$G_{27} = [2^3, 1583]$

$G_{28} = [2^7, 5, 29]$

$G_{29} = [3, 9067]$

$G_{30} = [5, 7, 17, 67]$

$G_{31} = [3, 5^2, 19, 41]$

$G_{32} = [2. 3^2, 67, 71]$

$G_{33} = [67, 1873]$

$G_{34} = [2^2, 45979]$

$G_{35} = [2, 7, 13, 1481]$

$G_{36} = [31, 12743]$

$G_{37} = [3, 7, 19, 1451]$

$G_{38} = [7, 47, 2579]$

$G_{39} = [2^2, 3, 173, 599]$

$G_{40} = [3^3, 67499]$

$G_{41} = [2^2, 667741]$

$G_{42} = [2^3, 17, 107, 269]$

$G_{43} = [37, 47, 3299]$

$G_{44} = [5^2, 336317]$

$G_{45} = [3^2, 47, 29131]$

$G_{46} = [2, 47, 192121]$

$G_{47} = [3^2, 53, 55487]$

$G_{48} = [2^4, 3^3, 13, 6907]$

$G_{49} = [2, 7, 359, 11311]$

$G_{50} = [5, 3851, 4327]$

$G_{51} = [257, 557, 853]$

$G_{52} = [11, 16268653]$

$G_{53} = [2^4, 3, 13, 420307]$

$G_{54} = [5, 7, 41, 277, 967]$

$G_{55} = [2^4, 3, 13, 902777]$

$G_{56} = [2^6, 3^2, 7, 13, 19, 829]$

$G_{57} = [7, 11, 73, 215261]$

$G_{58} = [67, 4271, 6197]$

$G_{59} = [5, 11, 47253079]$

$G_{60} = [2, 11, 367, 471749]$

$G_{61} = [3, 5, 43, 1361, 6359]$

$G_{62} = [2^2, 5^2, 83, 985679]$

$G_{63} = [2, 3, 23, 67, 1296781]$

$G_{64} = [3^4, 401, 541001]$

$G_{65} = [13, 67, 29567611]$

$G_{66} = [53, 67, 347, 30631]$

$G_{67} = [2^2, 23, 89, 293, 23057]$

$G_{68} = [7, 11581295567]$

$G_{69} = [2^2, 3^3, 5477, 200861]$

$G_{70} = [2^3, 1439, 2719, 5563]$

$G_{71} = [3^3, 9451749779]$

$G_{72} = [3^4, 509, 9071521]$

$G_{73} = [7, 78305416339]$

$G_{74} = [2, 11, 17, 37, 58052837]$

$G_{75} = [5, 7, 33638425649]$

$G_{76} = [2^3, 7, 30812193073]$

$G_{77} = [2, 3, 71, 251, 1999, 11831]$

$G_{78} = [577, 6423159217]$

$G_{79} = [3, 23, 78719502613]$

$G_{80} = [3^2, 29, 1409, 21646459]$

$G_{81} = [2^3, 5^3, 11666626519]$

$G_{82} = [17098272199297]$

$G_{83} = [2^3, 354667, 8831783]$

$G_{84} = [2^6, 47, 79, 154547209]$

$G_{85} = [3, 5, 11^2, 29654895079]$

$G_{86} = [8863, 8900188471]$

$G_{87} = [3, 7, 36671, 150122171]$

$G_{88} = [2, 3^3, 17^2, 99559, 109049]$

$G_{89} = [23, 47, 229707435499]$

$G_{90} = [2^2, 5, 2908183, 6256853]$

$G_{91} = [2, 13, 47, 67, 6514312699]$

$G_{92} = [5, 7, 47, 475177249661]$

$G_{93} = [3^2, 5^2, 111949, 45480553]$

$G_{94} = [7, 29, 24847, 332862911]$

$G_{95} = [2^2, 3^2, 7, 9764315316923]$

$G_{96} = [3^3, 67, 1993474574969]$

$G_{97} = [2^2, 372461, 3547442813]$

$G_{98} = [2^3, 67, 1109, 3803, 3426419]$

$G_{99} = [67, 4421, 38324345327]$

$G_{100} = [23, 1481, 116359, 4197533]$

$G_{101} = [3, 79, 1426699, 72111283]$

$G_{102} = [2, 137, 130418828295607]$

$G_{103} = [3, 56268347, 310250419]$

$G_{104} = [2^5, 3^2, 13, 19, 1078984681403]$

$G_{105} = [2, 367, 1973, 77676295693]$

$G_{106} = [5, 7, 37, 1273059829720 79]$

$G_{107} = [293, 1460423, 564649867]$

$G_{108} = [6793, 56053, 929976491]$

$G_{109} = [2^5, 3, 13, 23, 43, 420463693127]$

$G_{110} = [14741101, 51596041957]$

$G_{111} = [2^5, 3, 7, 13, 37, 641, 5379982483]$

$G_{112} = [2^7, 3^3, 5, 11, 13, 93133, 7098667]$

$G_{113} = [7, 37, 31477, 293679752479]$

$G_{114} = [7, 31, 37, 118453, 36894914 89]$

$G_{115} = [38831869, 132431919157]$

$G_{116} = [2, 5, 823, 1429, 6521, 98274679]$

$G_{117} = [3^2, 11, 347, 321536411315891]$

$G_{118} = [2^2, 4047079818514504489]$

$G_{119} = [2, 3^2, 11, 19, 6306521802364603]$

$G_{120} = [3^3, 11, 137, 1511, 565554084671]$

$G_{121} = [5, 13, 701, 1118384607318961]$

$G_{122} = [131, 158521469, 3596417509]$

$G_{123} = [2^2, 5, 5472760250570726723]$

$G_{124} = [5^2, 19, 53, 67, 449161, 211737133]$

$G_{125} = [2^2, 3, 7^2, 2657, 150480907381661]$

$G_{126} = [2^3, 19, 2266802199396721193]$

$G_{127} = [3, 19, 131, 6762666835182083]$

$G_{128} = [3^2, 150671099, 545756115407]$

$G_{129} = [67, 131, 163, 26474891, 28635853]$

$G_{130} = [2, 7^2, 47, 131, 97843, 26925262733]$

$G_{131} = [67, 34770990966547606513]$

$G_{132} = [2^3, 7^2, 23, 67, 2503, 2258131593659]$

$G_{133} = [2, 3, 7^2, 129049, 2130539, 61903367]$

$G_{134} = [11, 1543, 2069, 20883032459577]$

$G_{135} = [3, 29, 47, 2273, 1156386600395969]$

$G_{136} = [3^4, 271, 5347051, 134201661371]$

$G_{137} = [2^3, 5, 19, 47, 73, 79, 181, 619146587303]$

$G_{138} = [41, 47, 83, 251, 31859, 147409, 179453]$

$G_{139} = [2^3, 2350462613, 2636962710271]$

$G_{140} = [2^8, 149, 1905144936352967641]$

$G_{141} = [3^3, 1597, 1115129359, 2214966977]$

$G_{142} = [97, 1609149457890676792633]$

$G_{143} = [3^3, 5, 31, 54661253492865769793]$

$G_{144} = [2, 3^4, 7, 3257, 5237723, 17330402153]$

$G_{145} = [11, 149, 941, 2558189, 12453393493]$

$G_{146} = [2^2, 397, 1297, 61681, 1145611, 4947853]$

$G_{147} = [2, 5^2, 13, 149, 75583, 1441713696523357]$

$G_{148} = [149, 653, 1589682000730484960 9]$

$G_{149} = [3, 7, 2836033, 7211363, 5277991739]$

$G_{150} = [208337, 26635423, 598683880901]$

$G_{151} = [2^2, 3, 7, 1279, 45319036771844289659]$

$G_{152} = [3^2, 5^2, 7, 457, 2861, 22849163, 151653529]$

$G_{153} = [2^2, 12540119, 2084887930032176 51]$

$G_{154} = [2^3, 5^2, 73, 31117699, 33735796195253]$

$G_{155} = [5^4, 35940011346663158022289]$

$G_{156} = [2063, 4556269, 3502326047335783]$

$G_{157} = [3, 37, 67, 97, 6688097173882339 5809]$

$G_{158} = [2, 17, 19, 1052693, 1621031, 64143617407]$

$G_{159} = [3, 3019, 5533901, 2067617503661711]$

$G_{160} = [2^4, 3^3, 13, 151, 1153, 247337, 657661, 954923]$

$G_{161} = [2, 953, 26449, 311609, 381533, 37138603]$

$G_{162} = [29, 67, 6551, 11602427, 2208907364771]$

$G_{163} = [7, 68299203552137191591478887]$

$G_{164} = [19, 67, 173, 2909, 1674817, 653032840799]$

$G_{165} = [2^4, 3^2, 13, 67, 1201, 1259, 5081, 184649, 5771411]$

$G_{166} = [21323, 7058073343376839881 1811]$

$G_{167} = [2^4, 3^2, 13, 17, 4933, 14049974116528745173]$

$G_{168} = [2^6, 3^3, 5, 7, 13, 243461, 16887439284150059]$

$G_{169} = [173, 6883, 13397, 296977699271223931]$

$G_{170} = [7, 271, 39805763, 91949403486904739]$

$G_{171} = [7, 173, 365941, 22962222337479290039]$

$G_{172} = [2, 11, 107, 149, 173, 2457747329777322919 69]$

$G_{173} = [3, 17, 89, 227, 677, 3133345504645480095 1]$

$G_{174} = [2^2, 5, 160162164843958296574798492 1]$

$G_{175} = [2, 3, 31, 2711, 2967292327, 31375746117791]$

$G_{176} = [3^2, 47, 857, 1621, 6356149, 18420673952287]$

$G_{177} = [11, 13, 705138915061101420620329811]$

$G_{178} = [5, 311, 134269, 707801541747440836741]$

$G_{179} = [2^2, 11, 982338709007, 5010841312558231]$

$G_{180} = [11, 1171, 24642339402104094394766041]$

$G_{181} = [2^2, 3, 47, 281, 997, 52813, 557464256455827 59]$

$G_{182} = [2^3, 7, 15817, 769720824703853527359557]$

$G_{183} = [3, 5, 17, 19, 47, 359, 4383763, 27881693899102 87]$

$G_{184} = [3^3, 47, 53, 217731743119662316529174 93]$

$G_{185} = [5, 29, 32909, 10387690441, 43297665990113]$

$G_{186} = [2, 5^2, 41, 16638077, 27425747, 33624640358 29]$

$G_{187} = [7, 2341, 15187, 285470663, 6188541779273 3]$

$G_{188} = [2^3, 37, 32783, 69622010828242228951033 7]$

$G_{189} = [2, 3^2, 7, 43, 71, 25739289187559713310217797]$

$G_{190} = [7, 67, 309405470131697212373966980 93]$

$G_{191} = [3^2, 3779, 4729, 348011, 5314432533, 710940389]$

$G_{192} = [3^3, 89, 229, 2341, 24119491086233533018 6197]$

$G_{193} = [2^3, 23^2, 1489, 13363957, 54213311910525013 1]$

$G_{194} = [11, 607, 1171, 2341, 36575382053684624656 67]$

$G_{195} = [2^3, 43, 67, 2341, 3272303, 55570885653838578 7]$

$G_{196} = [2^6, 1019, 22048967704883142059314703 29]$

$G_{197} = [3, 67, 402808111, 2602885634144591260183]$

$G_{198} = [67, 71, 27391723, 2370302528274122968751]$

$G_{199} = [3, 5^3, 307, 607, 1971601, 2509313, 1309275415979]$

$G_{200} = [2, 3^2, 289573, 12727396523743285661234 5817]$

$G_{201} = [7, 31, 607, 2503, 2948952670692330599053507]$

$G_{202} = [2^2, 607, 1628814526159, 360299527190539687]$

$G_{203} = [2, 13, 73, 3823649, 232875749, 1235641393813019]$

$G_{204} = [173979434527, 17591381879176070679143]$

$G_{205} = [3, 5, 11, 61, 109, 26066303987, 156850197574469923]$

$G_{206} = [7, 175853, 1462229, 365214927246293360 9533]$

$G_{207} = [2^2, 3, 17, 163, 229, 431, 1013, 749587, 3865953716262641]$

$G_{208} = [3^5, 7, 461, 2789, 169369, 67753219, 562610699647]$

$G_{209} = [2^2, 5, 7, 179, 1142719704094 9, 72262294331976181]$

$G_{210} = [2^3, 3790962370199179303894941794183231]$

$G_{211} = [173, 45707, 2329567, 241129169057294301989 93]$

$G_{212} = [53, 113, 431, 443, 858479, 663572413673921976947]$

$G_{213} = [3^4, 2181869, 117832266527, 458439819472054 3]$

$G_{214} = [2, 5, 431, 12889, 67789, 3715450377507398279 5577]$

$G_{215} = [3^4, 431, 167842212489509, 34995373525580249]$

$G_{216} = [2^6, 3^5, 5, 13, 163, 421, 433223239134510181211 4457]$

$G_{217} = [2, 5^2, 61, 7193051, 200758990007092813856756569]$

$G_{218} = [31, 227, 5801, 158126410974191541104885385 73]$

$G_{219} = [9539, 698539657, 589293686 89, 2409218227523]$

$G_{220} = [7, 17, 37, 29753, 52817, 330557, 3999923, 151550582107]$

$G_{221} = [2^6, 3, 13, 271, 10901351, 58011716 53, 47498876529959]$

$G_{222} = [47, 727, 11793443650215 77, 739000901049153601]$

$G_{223} = [2^6, 3, 13, 67, 227, 17109, 44647, 11783212778 5400212597]$

$G_{224} = [2^8, 3^2, 13, 17, 3851, 717908437, 154379968839823 9897]$

$G_{225} = [7, 37, 227, 6977, 30011, 2015 8567, 37221460623199277]$

$G_{226} = [227, 9370373, 127818901787, 505327050466 12829]$

$G_{227} = [7, 37, 47, 1151, 46807, 2135162515660789770371776 9]$

$G_{228} = [2, 7, 37, 67, 95019938137 09, 8018390347415179 6237]$

$G_{229} = [3, 47, 101, 3037, 372850656827, 26819496422709121 09]$

$G_{230} = [2^2, 5, 47, 59, 67, 17057815900850749423963998338147]$

$G_{231} = [2, 3, 19, 67, 89, 661, 4839220261, 427207149683034203 29]$

$G_{232} = [3^3, 11, 10429, 13887366239, 31619941164864173 41867]$

$G_{233} = [13, 46147, 66509, 50007096815228598790296151 91]$

$G_{234} = [29, 3257, 562950840166 4113, 54994541907771 2023]$

$G_{235} = [2^2, 151, 3227417, 16993186324584 7, 1293736729985471]$

$G_{236} = [5^2, 25123472790323140642534222014282731153]$

$G_{237} = [2^2, 3^2, 11, 5352362129, 369317008399, 1175943964398667]$

$G_{238} = [2^3, 43, 9067, 3323120587, 161166362927, 807590085319]$

$G_{239} = [3^2, 7, 11, 32385733, 9253248101, 9520490018330700677]$

$G_{240} = [3^3, 5, 11, 282881, 1781317, 3872362102181978667697213]$

$G_{241} = [2955749, 100201241576641, 14338828928242531697]$

$G_{242} = [2, 41, 1013671, 764654837615197, 97923055019771929]$

$G_{243} = [20701027, 572512847, 76964609851786384679855 9]$

$G_{244} = [2^3, 7, 149427611, 12951017477, 123353751625046110117]$

$G_{245} = [2, 3, 5, 1450680512189, 450182938766532476512161509]$

$G_{246} = [7, 19, 6271, 8852411, 3889009322267441370830785337]$

$G_{247} = [3, 5, 7, 157, 419, 609246372665809650511328043064466 3]$

$G_{248} = [3^2, 5^2, 5281, 858126209, 26581876667, 2275450369422709]$

$G_{249} = [2^3, 71, 25717, 6187869543507950028908717193946771]$

$G_{250} = [593, 66667194089, 335081037146871587766917767 3]$

$G_{251} = [2^3, 19, 40852429, 312652698083494246994394783091 37]$

$G_{252} = [2^7, 131, 14947993, 113518444906788630459102193475 3]$

$G_{253} = [3, 19, 2017, 103993, 34878086856029513071441022692529]$

$G_{254} = [11^2, 19, 389, 514531, 1374761, 670058477479, 1441798002091]$

$G_{255} = [3, 31034661787, 962017798660209538791602538180 1]$

$G_{256} = [2, 3^3, 67, 229, 788351, 2009715779932137115440857249207]$

$G_{257} = [59, 131, 248909764811228406574713617138355141667]$

$G_{258} = [2^2, 7^2, 41, 17683, 19841582095781336932570756078130033]$

$G_{259} = [2, 13, 17, 71, 79, 101, 131, 23192013150949, 5431756307855268869]$

$G_{260} = [131, 46229025708979507452469585592661250568371]$

$G_{261} = [3^2, 5, 67, 24373, 319749462749603699, 377734662577716221]$

$G_{262} = [3931, 33090006731588519195307785979958667994 73]$

$G_{263} = [2^2, 3^2, 7^2, 67, 3680662229, 1148255397083, 38165326976349829]$

$G_{264} = [3^3, 19, 67, 7140126217448526 43, 1138454572110136978273]$

$G_{265} = [2^2, 7^2, 11, 977, 1321, 52721, 431097077, 1388913577, 466166021899]$

$G_{266} = [2^3, 7^2, 23, 2822667257, 3307538687, 712933634602677086959]$

$G_{267} = [5, 43, 601, 6806465146418988937751683846201588974 71]$

$G_{268} = [47, 883807, 3103031565889464569503003227929189893]$

$G_{269} = [3, 887, 12311401, 576628452942064259308176654337811 3]$

$G_{270} = [2, 1174793, 99116093, 381587069377, 3115488708493480523]$

$G_{271} = [3, 5, 37, 1117, 24957437, 3888357409456901, 6744503210284429]$

$G_{272} = [2^4, 3^2, 13, 31766061332258601078405112809969770323390 9]$

$G_{273} = [2, 47, 1171, 3319, 236701, 1007822505316682633360839044600 1]$

$G_{274} = [283, 42443, 3760821696925206047, 2827533927401201712 7]$

$G_{275} = [47, 163, 28484977, 857805197952177785823752035442179 7]$

$G_{276} = [5, 47, 73, 1117, 92570329, 154661150959584862278843866344 1]$

$G_{277} = [2^4, 3, 7, 13, 97, 523, 64575182233271, 280984380490927990658557]$

$G_{278} = [5, 157, 1117, 2685316121, 3660811829, 683620611625790987417]$

$G_{279} = [2^4, 3, 5^2, 13, 79, 103, 283, 1117, 32393099027, 6644108460896810460257]$

$G_{280} = [2^6, 3^4, 13, 17, 11121919, 9933165517437067770171214869180 11]$

$G_{281} = [283, 499, 32771, 12225701, 61739146589, 5310318023183878907]$

$G_{282} = [7, 283, 24573276170929979, 55845861190994843856760639]$

$G_{283} = [8689, 1891391, 567543709, 427164380727001417513410010 7]$

$G_{284} = [2, 7, 293, 1213, 4118996741, 180282281598071, 15803433077843093]$

$G_{285} = [3^3, 7, 17, 19, 2039, 8443, 2913846909959, 27945652341309947728079]$

$G_{286} = [2^2, 61, 585863, 945671, 92776991846755222444166112501103 3]$

$G_{287} = [2, 3^3, 17^2, 163, 2311, 31267563084692603309285289319894545493]$

$G_{288} = [3^4, 17, 149, 631, 3923, 6323, 83885917864248746594459412676139]$

$G_{289} = [13, 67, 293, 601, 20063274494283623, 12829950709918734627672 1]$

$G_{290} = [63199, 390503, 1325274516549103, 17691047702339769119573]$

$G_{291} = [2^2, 19, 23, 293, 1697, 81857640983, 11919301283014269548697095989]$

$G_{292} = [5, 11, 293, 4993129, 15778549097, 97889854498673026096746157 1]$

$G_{293} = [2^2, 3, 149, 98783777, 10312444741566958218975689921117465413]$

$G_{294} = [2^3, 67, 12953, 29365401787081, 13093344814347161652342226633]$

$G_{295} = [3, 149, 875227825026279253679424593540172849199764780 7]$

$G_{296} = [3^2, 7, 67, 149, 1601, 56943298218312757892880084372271726353 73]$

$G_{297} = [11, 67, 3457, 5233, 144737, 704614663, 61800737384981429283126 43]$

$G_{298} = [2, 5, 103, 757, 9209, 22273, 77006101892078413213968002756014949]$

$G_{299} = [11, 1229159, 308071279, 197178665881, 21975789140404162143097]$

$G_{300} = [2^3, 11, 601, 2251, 22219314581115055730128515294491366627707 1]$

$G_{301} = [2, 3, 7, 1093, 3617, 465960166649, 827412112959911, 6055874525584201]$

$G_{302} = [5^2, 31, 37, 41, 425378207, 1030450637, 11025224671196300520324904 1]$

$G_{303} = [3, 7, 89, 21467, 20754077482187270181912140664956309137993 51]$

$G_{304} = [3^3, 7, 9177941, 25556507, 9187051323153101, 299644506115443257]$

$G_{305} = [2^3, 127, 331, 12157955743, 64292008877, 338670860681, 2009004303941]$

$G_{306} = [23, 1609, 16223, 18067739, 18734360375566673, 1289873615530232947]$

$G_{307} = [2^3, 5^2, 17, 31, 1373, 4153, 6029, 43783, 297151, 81491205484643320763328 43]$

$G_{308} = [2^6, 15686309, 56081291856944221466100207736015992891060 1]$

$G_{309} = [3^2, 5^2, 29, 31, 1123, 4565999197, 25729216353788281, 30920130964652021]$

$G_{310} = [5^4, 19, 31, 499888813, 2360045671, 1682739366079, 1654728016044889]$

$G_{311} = [3^2, 3607, 19687, 326951, 37666409047, 2296803171749, 98041666078157]$

$G_{312} = [2, 3^5, 67575947, 89143981, 79848828678646978197656005160172 17]$

$G_{313} = [422431, 29347434605171774881, 30706299511319010350591454 1]$

$G_{314} = [2^2, 11, 47, 2257909, 11948224358991307980120675229091202753680 71]$

$G_{315} = [2, 7, 13, 192979, 4675309, 213773743119919663, 232927705635497967953]$

$G_{316} = [71, 239, 277, 3497227, 72897789577799600107452686251660505507 9]$

$G_{317} = [3, 1321, 2137, 20737315644436487659192900497781107059002885 51]$

$G_{318} = [17, 401, 168523333747, 917308976511989, 24424128992707658891669]$

$G_{319} = [2^2, 3, 41, 47, 73, 1201, 8963, 3604225858449127, 5759712062295320831392 69]$

$G_{320} = [3^2, 7, 23, 149, 7243, 33533, 65807206860823213, 160207528732691643533 27]$

$G_{321} = [2^2, 47, 191, 397, 1217, 11719, 1712077, 232767697127177441278362626158343]$

$G_{322} = [2^3, 7, 47, 67, 157, 1613, 26590168093030833160954161119686107595926 29]$

4 Tribonacci Number Factorizations

$T_3 = [2]$

$T_4 = [2^2]$

$T_5 = [7]$

$T_6 = [13]$

$T_7 = [2^3, 3]$

$T_8 = [2^2, 11]$

$T_9 = [3^4]$

$T_{10} = [149]$

$T_{11} = [2, 137]$

$T_{12} = [2^3, 3^2, 7]$

$T_{13} = [3^2, 103]$

$T_{14} = [5, 11, 31]$

$T_{15} = [2^6, 7^2]$

$T_{16} = [2^3, 7, 103]$

$T_{17} = [103^2]$

$T_{18} = [13, 19, 79]$

$T_{19} = [2, 5, 37, 97]$

$T_{20} = [2^2, 3, 5501]$

$T_{21} = [5, 7, 3469]$

$T_{22} = [3^5, 919]$

$T_{23} = [2^3, 51343]$

$T_{24} = [2^2, 188869]$

$T_{25} = [3^2, 181, 853]$

$T_{26} = [3^2, 199, 1427]$

$T_{27} = [2, 5, 470077]$

$T_{28} = [2^4, 7, 17, 19, 239]$

$T_{29} = [23, 47^2, 313]$

$T_{30} = [5^2, 37, 103, 307]$

$T_{31} = [2^6, 5, 7^2, 47, 73]$

$T_{32} = [2^4, 7, 883483]$

$T_{33} = [3, 53, 103, 11113]$

$T_{34} = [103^2, 139, 227]$

$T_{35} = [2, 3^5, 11, 41, 53^2]$

$T_{36} = [2^2, 233, 317, 3833]$

$T_{37} = [7, 11, 163, 263, 631]$

$T_{38} = [3^3, 3539, 40093]$

$T_{39} = [2^3, 3^3, 13, 577, 4349]$

$T_{40} = [2^2, 9133, 354763]$

$T_{41} = [167, 142739687]$

$T_{42} = [11, 47, 1877, 45181]$

$T_{43} = [2, 40320889337]$

$T_{44} = [2^3, 7, 37, 71584631]$

$T_{45} = [5, 47, 149, 2339, 3331]$

$T_{46} = [3, 47, 647, 5500283]$

$T_{47} = [2^7, 7^2, 103, 1428613]$

$T_{48} = [2^3, 3^4, 7, 53, 1213, 5821]$

$T_{49} = [199, 15689304167]$

$T_{50} = [5^2, 41, 103, 163, 333701]$

$T_{51} = [2, 3^2, 53, 103^2, 1043597]$

$T_{52} = [2^2, 3^4, 5^2, 13, 53, 31328771]$

$T_{53} = [7, 163, 31316187787]$

$T_{54} = [163, 257, 647, 1229, 1973]$

$T_{55} = [2^3, 13, 19, 23, 79, 1451, 23203]$

$T_{56} = [2^2, 13, 479, 59539, 149921]$

$T_{57} = [29531, 13847588627]$

$T_{58} = [5, 1279, 117614590727]$

$T_{59} = [2, 3, 139, 70229, 23619389]$

$T_{60} = [2^5, 7, 31, 366429917899]$

$T_{61} = [3^5, 5^3, 11, 277, 3313, 15263]$

$T_{62} = [5, 13, 19, 199, 3502510858\!9]$

$T_{63} = [2^7, 7^2, 532949, 4736497]$

$T_{64} = [2^5, 3^2, 7, 59, 103, 7151, 332393]$

$T_{65} = [3^2, 199, 2990558270764\!7]$

$T_{66} = [199, 495044179630231]$

$T_{67} = [2, 103, 2011, 437388591317]$

$T_{68} = [2^2, 61, 103^2, 128745455933]$

$T_{69} = [7, 136956013, 639390949]$

$T_{70} = [19, 857, 69240572393303]$

$T_{71} = [2^3, 37, 157, 4462242336008\!3]$

$T_{72} = [2^2, 3, 191, 36010153, 46211959]$

$T_{73} = [1259, 5572084228120051]$

$T_{74} = [3^5, 13, 61, 1619, 38321, 1079269]$

$T_{75} = [2, 47^2, 73, 5441, 31627, 427619]$

$T_{76} = [2^3, 5, 7, 44789, 165811, 20991767]$

$T_{77} = [3^3, 29, 47, 879391, 2480844481]$

$T_{78} = [3^3, 4673, 1170391273432363]$

$T_{79} = [2^6, 7^2, 17, 131, 349483, 111280159]$

$T_{80} = [2^3, 7, 751, 10559, 1124964332707]$

$T_{81} = [5, 103, 33060151, 53966827549]$

$T_{82} = [43, 39302478475197493579]$

$T_{83} = [2, 5, 37, 211, 6600791, 6031949213]$

$T_{84} = [2^2, 103, 1387682351667362444 1]$

$T_{85} = [3, 7^3, 53, 103^2, 2958827. 6142589]$

$T_{86} = [19341322569415713958901]$

$T_{87} = [2^3, 3^4, 53^2, 563, 1607, 23159, 932749]$

$T_{88} = [2^2, 47, 3389651, 4879871, 21040879]$

$T_{89} = [5, 24069357314251626329837]$

$T_{90} = [3^2, 59, 3109, 134081450289876211]$

$T_{91} = [2, 3^2, 47, 163, 1636667, 1803910939031]$

$T_{92} = [2^4, 5^2, 7, 17, 47, 199, 3203, 57593. 9117973]$

$T_{93} = [5, 11, 4914769, 5095261438808617]$

$T_{94} = [645626791, 3923739552074471]$

$T_{95} = [2^6, 7^3, 13, 17, 353, 10328047, 263434033]$

$T_{96} = [2^4, 7, 17, 587, 3719. 2061813501552821]$

$T_{97} = [15762679542071167858843489]$

$T_{98} = [3, 103, 587, 13177, 97440757, 124487491]$

$T_{99} = [2, 2971, 628800521, 14271952545419]$

$T_{100} = [2^2, 3^6, 53, 67, 179, 3217. 131561. 125028473]$

$T_{101} = [7, 43, 103, 1447, 867829. 4633630620551]$

$T_{102} = [103^2, 150436394809, 207897801781]$

$T_{103} = [2^3. 3^2, 19. 43, 53, 1125833, 173869061658839]$

$T_{104} = [2^2. 3^2. 53. 163. 547. 550999403. 11974895201]$

$T_{105} = [109, 149, 25095481. 5065441495731209]$

$T_{106} = [11, 97, 647. 30471629. 180513906798077]$

$T_{107} = [2. 5, 163. 449, 503, 1303, 1456056190239779 9]$

$T_{108} = [2^3. 7^2, 13, 131. 163. 3237456763. 36465454279]$

$T_{109} = [11, 89, 499. 7207. 12073. 122279. 1545885887]$

$T_{110} = [11. 6203, 12377, 62723. 8204197983421 63]$

$T_{111} = [2^8, 3, 7^3, 13. 499, 2991532669. 156361298 09]$

$T_{112} = [2^3. 5, 7^2, 13. 12457. 38723. 448367. 26678178467]$

$T_{113} = [3^6. 263, 715909, 1513819, 13013858830 81]$

$T_{114} = [5, 41, 1831. 7650095473. 173205438780923]$

$T_{115} = [2, 103, 199. 397, 42491, 1464503. 903279195953]$

$T_{116} = [2^2, 3^4, 19, 1820089. 150168290372918595521]$

$T_{117} = [3^4, 7, 23, 31. 2207. 170557. 203364173359152 19]$

$T_{118} = [11, 13, 103, 421, 9179372660498582766532 73]$

$T_{119} = [2^3, 19, 103^2, 282013604617, 23021292668237]$

$T_{120} = [2^2, 5, 19, 176250983, 287509435551556989977]$

$T_{121} = [47^2, 66719903, 240306527543021770367]$

$T_{122} = [3947, 135913, 41134434977, 2952109566463]$

$T_{123} = [2, 5^2, 23, 29^2, 47, 311, 421, 27803, 797021, 908489629]$

$T_{124} = [2^6, 3, 5, 7, 11, 17, 183283, 956823770517321050251]$

$T_{125} = [73, 555253882317001840315609679 1047]$

$T_{126} = [3^4, 92040482890696286103775567567 21]$

$T_{127} = [2^8, 7^2, 773, 92669, 3583847, 425808056670847]$

$T_{128} = [2^6, 7, 199, 397, 999763, 712761342553148993 47]$

$T_{129} = [3^2, 773, 354832421, 187917376253831 0081921]$

$T_{130} = [3^2, 13, 883, 2243, 393137357, 93657277405620793]$

$T_{131} = [2, 199, 397, 487, 10343, 66083, 81749, 3649974754717]$

$T_{132} = [2^2, 71, 103, 199, 397, 392531, 133064033, 239125988033]$

$T_{133} = [7, 12501361, 606673011568892522705725073]$

$T_{134} = [37, 47, 73, 563, 13487, 29885 4181, 338964272222207]$

$T_{135} = [2^3, 103, 56263, 1318278178 9, 2938676217202767 83]$

$T_{136} = [2^2, 29, 103^2, 421, 2509719871, 2540196251909372 21]$

$T_{137} = [3, 47, 53, 1291, 70061, 102079, 8805901483909145041]$

$T_{138} = [5^2, 19, 47, 5005710876038436260639911995204 17]$

$T_{139} = [2, 3^5, 29, 53^2, 421, 761, 325691, 232231711, 2142528631]$

$T_{140} = [2^5, 7, 29, 421, 42356767, 851966537, 153220595225383]$

$T_{141} = [5238048231571481, 1327504697696897810039]$

$T_{142} = [3^2, 13647, 1048288606325 99, 9931994629907 0009]$

$T_{143} = [2^6, 3^2, 5, 7^2, 883, 5791, 9131, 3456558134338181245283]$

$T_{144} = [2^3, 7, 47983577, 1610175607403070211956592 1311]$

$T_{145} = [5, 11, 163, 167, 6189216 19181, 8588133531351740 0951]$

$T_{146} = [883, 330413093067421 7, 5016892811039096 6917]$

$T_{147} = [2, 11, 883, 51343, 315866621, 78012096443586463303 43]$

$T_{148} = [2^2, 19, 23, 291503, 2933619619181, 3312523173723116 09]$

$T_{149} = [7, 31, 103, 499, 8165891146623321910621562906180 03]$

$T_{150} = [3, 1153, 120087827, 267314 4953, 1508615249692077 413]$

$T_{151} = [2^3, 5^2, 13, 1185022636917874818639479892026218051]$

$T_{152} = [2^2, 3^5, 11, 53, 103, 4658693790 07, 208107444282587628569]$

$T_{153} = [103^2, 7827857, 1440608897398 67, 871235285036531]$

$T_{154} = [5^3, 31, 3169486339 1, 1560945477821221585490681329]$

$T_{155} = [2, 3^3, 5^2, 53, 3613, 1364018211520097964747625710247 31]$

$T_{156} = [2^4, 3^3, 7, 53, 61, 347, 191174624443741714520733860478 49]$

$T_{157} = [8273, 4473881, 743931493, 4332277415795509059739]$

$T_{158} = [43, 163, 199, 402379, 39093239082574982595282216709 3]$

$T_{159} = [2^6, 7^2, 2154434329, 597291353438389831552646821 91]$

$T_{160} = [2^4, 7, 1823, 3635300318887417613248523195303451257]$

$T_{161} = [163, 1336211, 626804033220238975770055756445524 1]$

$T_{162} = [163, 499, 308713571991994434719903127454799363 77]$

$T_{163} = [2, 3, 4451568481189, 172913542036202063657927921791]$

$T_{164} = [2^2, 13, 3493573, 20743516563387661, 2254172832003073601]$

$T_{165} = [3^4, 7, 227, 499, 278925269849453, 872157419412350041213]$

$T_{166} = [89, 103, 499, 628223054112168881177970297570956035 3]$

$T_{167} = [2^3, 13, 47^2, 30675237779, 750022163612925086876539554 1]$

$T_{168} = [2^2, 3^2, 13, 3613, 1744516783937, 32957295576406327144886719]$

$T_{169} = [3^2, 5, 47, 103, 257, 5701, 3374942944725827, 165993767182604131]$

$T_{170} = [103^2, 6722042627, 46117282599047854123553709683 27]$

$T_{171} = [2, 11, 193, 257, 3613, 170197277, 90147810269308700635360 0819]$

$T_{172} = [2^3, 7, 113, 3613, 6629593, 1791300831703661, 4097778512449147]$

$T_{173} = [181, 9649, 17581, 8300743138823137, 8029089904016062591]$

$T_{174} = [5, 13, 61736743, 28302615823792033, 33140079633413042879]$

$T_{175} = [2^7, 7^2, 17, 19, 1459, 23421935153028432966895125885421 56553]$

$T_{176} = [2^3, 3, 5, 7, 193, 141403, 51685627223221, 1074659969790839014 7789]$

$T_{177} = [269, 17167, 966319, 52482959781094780031854796465987 41]$

$T_{178} = [3^5, 72024767017462705429, 24611995832245568716427 27]$

$T_{179} = [2, 269, 20061562860161, 73406982725482768604014315885 57]$

$T_{180} = [2^2, 47, 102200027753, 75844657164197467149225299974705 33]$

$T_{181} = [3^2, 7, 73, 199, 8885093563, 32961336096774913969749684692389]$

$T_{182} = [3^2, 5, 19, 659, 93851. 436013, 21381731445670690148479068574607]$

$T_{183} = [2^3, 31, 47, 103, 192325099. 39269869485172889819388116999860 27]$

$T_{184} = [2^2, 47, 107, 34519. 74544593, 8479713149, 3799570328120350592047]$

$T_{185} = [5^2, 772842937, 15876319201623823480212562109253998392 1]$

$T_{186} = [5, 13, 101, 103, 2039. 138923, 157933, 23003341, 8107806585125246784979]$

$T_{187} = [2, 103^2, 149, 901215439. 36421734424661978736744119918746 27]$

$T_{188} = [2^5, 7, 17, 331, 215920183, 2785465091, 25177552504566598520964329]$

$T_{189} = [3, 37, 53, 1237, 1789, 1088569, 386945512687, 6401683242362044941211]$

$T_{190} = [19, 2816533939537. 12065899414024972819110861843996942 83]$

$T_{191} = [2^7, 3^5, 7^2, 17, 53^2, 253769, 6662177, 107689668319, 8962640474594707]$

$T_{192} = [2^5, 7, 17, 149, 461, 1416382571, 12317412613457411, 47867746237096649]$

$T_{193} = [1663, 24159301385304683699033537181122067545807230 9183]$

$T_{194} = [3^3, 37, 199, 99146251, 1758438161218727423, 21320829766797385133]$

$T_{195} = [2, 3^3, 1181, 129208412028469135601, 16494572393678158906228735 9]$

$T_{196} = [2^2, 3989351, 177982063729, 246488232401, 3571003552995575867119]$

$T_{197} = [7^2, 199, 1181, 68716829, 350202467, 243824713427, 68047990817853169]$

$T_{198} = [199, 194727479, 218244606869284104178873796965412831809069]$

$T_{199} = [2^3, 163, 879175576943, 135681324754497528523473803320789500 31]$

$T_{200} = [2^2, 5, 103, 683, 443467, 458536058214365899408485967525722015474 01]$

$T_{201} = [2297, 2309513, 42851351, 3188308483, 529547891017, 137107327671221]$

$T_{202} = [3, 1237, 1230167, 731817230415097, 289710312536305104864837093 97]$

$T_{203} = [2, 11, 103, 33923, 60549898967, 382475371367436774955849381333548 89]$

$T_{204} = [2^3, 3^4, 7, 53, 103^2, 220353332094043108733, 582609433233864737166533]$

$T_{205} = [5^2, 311, 1237, 626180941616393781410607967281155375180014226 53]$

$T_{206} = [1237, 375465408440664244485 47, 238495261525858312745083325 03]$

$T_{207} = [2^6, 3^2, 5^2, 7^3, 13, 53, 311, 2442227, 61487816309, 128190992082775795948983 41]$

$T_{208} = [2^3, 3^2, 7, 53, 61, 930113, 247255703072568750328686351030531150820318 7]$

$T_{209} = [1752121324867, 393372088181235735933826632037254501948402 7]$

$T_{210} = [5717, 221742514433232121296172023049336650488746509633125 3]$

$T_{211} = [2, 7523, 154969255311564399364534754594640825391581109238398 7]$

$T_{212} = [2^2, 163, 3833, 3866028776338489, 5853639490746221, 758295672976518553]$

$T_{213} = [5, 7, 47^2, 11841629, 646738415641, 133217604963293989586084296708729]$

$T_{214} = [1450824677533516614381305019377544205840960000831839926 29]$

$T_{215} = [2^3, 3, 47, 163, 91163, 197063, 562256590267, 143684334349981678632323997623]$

$T_{216} = [2^2, 5^4, 11, 163, 1307, 352097, 593629, 400811802481356633385740175932471139]$

$T_{217} = [3^6, 5, 29, 103, 983, 3457, 2079713, 17098931, 686128162451185552188408833923]$

$T_{218} = [71, 1307, 178928191210239113281519508292891471595409815421823 53]$

$T_{219} = [2, 11, 31, 383, 1999, 584879242871511645290950195799547284677321437122 9]$

$T_{220} = [2^4, 3^2, 7^2, 11, 13, 17, 79, 103, 167113, 572521, 27705452724419299, 15155068255115773093]$

$T_{221} = [3^2, 31, 103^2, 102130801039599684257, 341763475504899027910137204810 97]$

$T_{222} = [61, 6433069, 387367468106602889345821 1, 125008427540378607903980 03]$

$T_{223} = [2^6, 7^3, 13, 19, 151, 24294461, 1845774809, 95197588709014735009591742324560 3]$

$T_{224} = [2^4, 7^2, 13, 199, 5417, 585110702277807468609592299596077167172196743216 9]$

$T_{225} = [101, 2381, 491674905701379909735606468046031599293476006944038529]$

$T_{226} = [47, 307, 1367, 3917, 5417, 289129, 333921741058943, 538213702611486057768092 9]$

$T_{227} = [2, 101, 1255169, 9855037, 3213413519, 1147703623571, 434059730969216969782 61]$

$T_{228} = [2^2, 3, 11, 1367, 3944407, 381996887731, 270597108515545781070564118005865058 3]$

$T_{229} = [7, 47, 146936934209625035071683 59, 279917283065375826451656570114329]$

$T_{230} = [3^6, 13, 41, 43, 47, 21211, 452948045667779, 329894780863693432095228924632909]$

$T_{231} = [2^3, 5, 37, 8821, 14221, 246573159081573299011236743087406361103240532174 83]$

$T_{232} = [2^2, 109, 2003, 3259, 4938403, 599054851825373874095729872993832297121566081]$

$T_{233} = [3^4, 30094792077551, 1913482243750249, 332011829400811093890160335976 7]$

$T_{234} = [3^4, 11, 103, 179, 449, 2341, 57681386351, 285991021754671416256957692789546321 53]$

$T_{235} = [2, 757, 44609051, 420160335592369, 184624501822102163995128088794579718 3]$

$T_{236} = [2^3, 5, 7, 19, 8553055613, 2888432618494 1, 7331747871262850560935650544166236 1]$

$T_{237} = [103, 523, 4849189, 9701971, 6993331851425953243994371569053992149091027 7]$

$T_{238} = [5, 103^2, 1439, 16319, 80222383543, 3262174781708041233628161439385654978291]$

$T_{239} = [2^9, 7^2, 19, 257, 360977, 19052829865079305423, 7116381715714262319352684787 63]$

$T_{240} = [2^3, 7, 19, 1439, 10007, 220903, 476299, 99233957, 6893702699703187277469447658130323]$

$T_{241} = [3, 53, 29033, 35083, 340381, 1082597, 23488527506621, 1447025636122749696282435833]$

$T_{242} = [13, 3259, 1888177756869869, 5218684588701509, 8936505195145325505199305431]$

$T_{243} = [2, 3^4, 53^2, 149, 16447, 20543, 1205085877, 2485576977089597927972916085784431173 53]$

$T_{244} = [2^2, 5, 611531, 708031, 2598415403, 5609044822478709127922555425974700880178 01]$

$T_{245} = [7, 31, 1051, 715451089, 1422673634539189387288067348110515300509306575068 13]$

$T_{246} = [3^2, 107, 7144979, 142873883, 5630303113087556263, 7714099876291595905618219537]$

$T_{247} = [2^3, 3^2, 5^2, 71, 199, 397, 49123, 2880739, 4323412073, 1271325350452769908166238474124997 79]$

$T_{248} = [2^2, 5, 59, 757, 26414473, 6121758251058516953288977472938358932730219892380323]$

$T_{249} = [229, 72923, 18444700295576992679069, 8625292740751547423416723449573783 07]$

$T_{250} = [21713, 1434143, 39728783, 591251303, 6680442605224351019374317658884079716 11]$

$T_{251} = [2, 103, 269, 757, 347561, 1329862577, 3683103075071389, 1258569743351783521414150180 3]$

$T_{252} = [2^7, 7, 257, 757, 9483240702940758173163706928920629556775681205429932618641]$

$T_{253} = [163, 1621, 44904517661, 16517415169537, 1551459667099015775048148915941483899 7]$

$T_{254} = [3, 103, 13207576049, 4104148923224173748253158 33, 3338781225540657787526378261]$

$T_{255} = [2^9, 7^2, 11, 103^2, 157, 257, 100777472857, 8639979299964127079465483416453402855818 79]$

$T_{256} = [2^7, 3^5, 7, 53, 257, 7510109, 8494191422025769466749277646744052632756034587212 17]$

$T_{257} = [11, 599, 5807, 1072849, 6675138989, 8056842898278438959659, 1576180176889085645549 3]$

$T_{258} = [19, 421, 1097, 40627, 398323517, 110549218091023, 4077009709921641736723962730294159]$

$T_{259} = [2, 3^2, 23, 47^2, 53, 9067, 54185707, 167065039, 2958925480744999930101365205795166861709 3]$

$T_{260} = [2^2, 3^2, 53, 199, 397, 324179, 6510463, 13665828191, 13164974047, 3782811178477321201261612459]$

$T_{261} = [7, 47, 107, 3635837629, 515400241679, 6036844708621127334909401459747101247378879]$

$T_{262} = [5, 11, 113, 966224141, 770915915449, 1582202959394270055658670928533984413948544 47]$

$T_{263} = [2^3, 13, 29^2, 149, 199, 397, 421, 251879, 4819448408233, 2560395682976617958121124698968305651]$

$T_{264} = [2^2, 23, 199, 269, 397, 12143, 16618951337, 75582339793, 5335766433627191, 1557243991029464 2019]$

$T_{265} = [401, 142537, 7973785803500173539004442541355297259204918305628821148800 6457]$

$T_{266} = [59, 163, 1621, 5377288829646695080481838592224161499021681837204496529360358 89]$

$T_{267} = [2, 3, 5, 211, 269, 401, 4297, 796363, 1269049, 18335497, 2835878178236099903348305780704929218 67]$

$T_{268} = [2^3, 7, 19, 103, 269, 1451, 14563, 13053608819, 86200207790332407905 57, 4045727065808 3181729081631]$

$T_{269} = [3^5, 5, 163, 1531, 1621, 139060457, 7631476750684528285242339571118104590762918033742 13]$

$T_{270} = [83, 163, 1621, 43745668491734764675424703935947108002806840215392302407474080 13]$

$T_{271} = [2^6, 7^2, 17, 103, 167, 215500014279589033692239581357 1, 8929090774834545050731748616670 7]$

$T_{272} = [2^3, 3^3, 7, 47, 103^2, 4304855295976823820508012774537554128759024558070592594938182 51]$

$T_{273} = [3^3, 43, 11096033982731761369, 4633744693786047358321227316159411989232246734907 3]$

$T_{274} = [656662324733, 167201220481005431044204828189041743907944529858431371008650 9]$

$T_{275} = [2, 5, 47, 499, 2153, 195541818413, 689755208825823681262086 7, 2965199476846943644963251287]$

$T_{276} = [2^2, 13, 29, 47, 421, 587, 10667, 12839903, 45347749, 1893130935047494123, 1803518531792141566155342 1]$

$T_{277} = [7, 73, 419, 499, 883, 3529, 25579, 2888443, 144639600162373, 1901096387248061, 1010051063282448283]$

$T_{278} = [5^2, 23, 131, 238729, 6987712010863830618813104288745024547621678932612002117964708 97]$

$T_{279} = [2^3, 5, 13, 29, 373, 419, 421, 410482043, 1939598501, 1941528019947101, 1506859646749004297082498377 9]$

$T_{280} = [2^2, 3, 13, 29, 421, 381630590537143125820730291258197, 584830929194866319428524760361704 87]$

$T_{281} = [11, 89, 1471, 527179, 994822531, 2451485999, 162936476454084494891, 2591676738131165186145 29]$

$T_{282} = [3^4, 97, 1063, 4349, 12983, 21176420407, 17180344310802767869, 8381817392617515006717679030 27]$

$T_{283} = [2, 67, 83, 53089, 31363552115941161768264 61, 1428281646251179359787597120274078689172 93]$

$T_{284} = [2^4, 7, 17, 140303681213, 1821125887292970373621823288803274761721969887247854712222281]$

$T_{285} = [3^2, 103, 907, 16759, 3446353116259, 1249788757769377640240900 9, 1474327315345976968352063383]$

$T_{286} = [3^2, 13, 83, 100357, 25922851595307907183, 6514490676068405180289514420868053664194526 4509]$

$T_{287} = [2^6, 7^2, 17, 83, 461, 64969, 267129133073161, 8550526914006643581812826774467637564930751 1501]$

$T_{288} = [2^4, 7, 17, 103, 3637, 5483, 433541604361, 3283804213249164728252051461625752537283662998889243]$

$T_{289} = [103^2, 587, 3167, 33604371735367, 1545136319480648988687330433722792259197190796839978 7]$

$T_{290} = [113, 199, 883, 25579, 7283711, 42522594990461329, 1197359013237690621604384024945859944436 89]$

$T_{291} = [2, 43, 277, 72341, 173861, 330337085139861201362734013 51, 3500256321078449740910461860468697]$

$T_{292} = [2^2, 139, 587, 47661208791337, 2773476986087953624 09, 1476960952324404258227912119384941019 7]$

$T_{293} = [3, 5^2, 7, 53, 151, 311, 587, 883, 997, 25579, 151687587022961395950916819, 4473226531030874684991167250 7]$

$T_{300} = [2^3, 5, 7, 7283, 11706181, 3496097191499444395353458168789778742630407426331248368476633650 17]$

$T_{301} = [67, 641, 907, 1519423, 763221981941, 1912314865569765345401638 1, 1777000010652458763252674081]$

$T_{302} = [19^2, 61, 103, 907, 35497462241, 129115624675459, 2994389597093124827458897539269590367463052051]$

$T_{303} = [2^7, 7^2, 15313, 138251, 3552163928066467552690 9, 1101001196289977412376416269900364297065472 49]$

$T_{304} = [2^3, 7, 43, 251, 3853, 6781, 331094403906372522811, 3916998371998831069243 9, 4663764997693105932684 97]$

$T_{305} = [47^2, 103, 1021907, 48669725903, 90106150091, 3087030750436094393 9, 5581047081410908195970575219]$

$T_{306} = [3, 5^2, 103^2, 311, 38333, 2637496471, 1291525354403912557616710431403404414539835580516589501 7623]$

$T_{307} = [2, 43, 47, 163, 1213, 43638705361, 320786404427, 5312276260642440212935711951089114030285106775 8729]$

$T_{308} = [2^2, 3^5, 43, 53, 89, 373, 110807, 2154603197, 27459028327822177, 2267428134173209210455560694433540135177]$

$T_{309} = [5^3, 7^2, 311, 379, 157427, 22748140948692439075467504 23, 7776447477131893832899178734092352167923]$

$T_{310} = [5^2, 19, 257, 311, 47871367, 1063874411366059629 17, 1912529726187626551970550758037954638482978837 1]$

$T_{311} = [2^3, 3^3, 53, 307318728467, 85359333376469, 1165946461859510409249 7, 1942507391364352063057971325927]$

$T_{512} = [2^8, 7, 23, 257, 727, 3613, P_{122}]$

$T_{513} = [37, 773, 1543, 1785689, P_{128}]$

$T_{514} = [3, 31, 1543, 432311251, C_{122}]$